NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

EXTENDING SIMPLE NETWORK MANAGEMENT PROTOCOL (SNMP) BEYOND NETWORK MANAGEMENT: A MIB ARCHITECTURE FOR NETWORK-CENTRIC SERVICES

by

James Gateau

March 2007

Thesis Advisor:	Alex Bordetsky
Second Reader:	Dan Dolk

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** March 2007	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE: Extending Simple Network Management Protocol (SNMP) Beyond Network Management: A MIB Architecture for Network-Centric Services			**5. FUNDING NUMBERS**
6. AUTHOR(S) Gateau, James B.			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000			**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A			**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.			**12b. DISTRIBUTION CODE**

13. ABSTRACT (maximum 200 words)

The promise of the Global Information Grid (GIG) includes connecting sensors, shooters and decision-makers who may not be physically co-located in a manner efficient for combat employment, decision-making and information sharing. Current information architecture strategies, such as Network-Centric Enterprise Services have started down one path, requiring the implementation of a Service Oriented Architecture (SOA) and all the requisite underpinnings thereof. These are, for an organization the size of the DoD, a very large problem set in and of themselves. An additional unfortunate side effect of choosing a conventional SOA as the backdrop for the GIG is that only those devices capable of running an entire web server/database stack are able to participate in the architecture, effectively excluding computationally constrained devices. Additionally, the connectivity requirements in a conventional SOA restrict participation by bandwidth-constrained and intermittently connected entities. This thesis investigates one possible solution, utilizing SNMP as the language and mechanism for sharing data between disparate systems. Specific decision-support MIBs will be developed to allow transmission of decision-specific information in both push (TRAP/SET) and pull (GET) directions.

14. SUBJECT TERMS SNMP, Network Services, Hypernodes, VIRT, Smart Push, Smart Pull, GIG, FORCEnet			**15. NUMBER OF PAGES** 183
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

EXTENDING SIMPLE NETWORK MANAGEMENT PROTOCOL (SNMP) BEYOND NETWORK MANAGEMENT: A MIB ARCHITECTURE FOR NETWORK-CENTRIC SERVICES

James B. Gateau
Lieutenant Commander, United States Navy
B.S., Rensselaer Polytechnic Institute, 1996

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN INFORMATION SYSTEMS AND OPERATIONS

from the

NAVAL POSTGRADUATE SCHOOL
March 2007

Author:	James Gateau
Approved by:	Dr. Alex Bordetsky Thesis Advisor
	Dr. Dan Dolk Second Reader
	Dr. Dan Boger Chairman, Department of Information Sciences

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

The promise of the Global Information Grid (GIG) includes connecting sensors, shooters and decision-makers who may not be physically co-located in a manner efficient for combat employment, decision-making and information sharing. Current information architecture strategies, such as Network-Centric Enterprise Services have started down one path, requiring the implementation of a Service Oriented Architecture (SOA) and all the requisite underpinnings thereof. These are, for an organization the size of the DoD, a very large problem set in and of themselves. An additional unfortunate side effect of choosing a conventional SOA as the backdrop for the GIG is that only those devices capable of running an entire web server/database stack are able to participate in the architecture, effectively excluding computationally constrained devices. Additionally, the connectivity requirements in a conventional SOA restrict participation by bandwidth-constrained and intermittently connected entities. This thesis investigates one possible solution, utilizing SNMP as the language and mechanism for sharing data between disparate systems. Specific decision-support MIBs will be developed to allow transmission of decision-specific information in both push (TRAP/SET) and pull (GET) directions.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

I. INTRODUCTION 1
  A. STRUCTURE OF THIS THESIS 2
    1. Chapter II: The GIG and FORCEnet 2
    2. Chapter III: Push, Pull and VIRT 2
    3. Chapter IV: O-I-T and TMN 3
    4. Chapter V: SNMP 3
    5. Chapter VI: ASN.1 and MIBs 3
    6. Chapter VII: Hypernodes and Their MIBs 3
    7. Chapter VIII: The TNT-MIO Experiments 4
    8. Chapter IX: Experimental Method and Results 4
    9. Chapter X: Crafting MIBs 4
    10. Chapter XI: Extensions and Future Research 4

II. THE GIG AND FORCEnet 5
  A. THE GLOBAL INFORMATION GRID (GIG) 5
    1. Definition 5
    2. GIG Services 7
  B. FORCENET 9
    1. Definition 9
    2. Functional Concept 10
    3. Capabilities 11
    4. Architecture 11

III. PUSH, PULL AND VIRT 13
  A. ON THE NATURE OF INFORMATION, BRIEFLY 13
  B. INFORMATION PUSH 15
  C. INFORMATION PULL 18
  D. VIRT 19
  E. BANDWIDTH AND RESOURCE IMPLICATIONS 23

IV. O-I-T AND TMN 25
  A. ORGANIZATION, INFORMATION AND TECHNOLOGY 25
    1. Layered Models 25
    2. A New Approach 28
      a. The Physical Layer 29
      b. The Technological Layer 30
      c. The Information Services Layer 30
      d. The Decision Support Layer 31
      e. The Cognitive Layer 31
      f. Task 32
      g. Organizations in the Model 32
    3. Organization – Information – Technology 32
  B. TELECOMMUNICATIONS MANAGEMENT NETWORK (TMN) 33

1. TMN Service Architecture 34

V. SNMP 39

A. HISTORY AND ARCHITECTURE 39

B. POLLING AND ALERTING 43

1. The `get` Family of Operations 45

a. Security Concerns 46

2. The `trap` Family of Commands 48

3. The `report` Command 49

C. SNMP'S BROADER APPLICABILITY 49

VI. ASN.1 AND MIBS 51

A. ABSTRACT SYTAX NOTATION 1 (ASN.1) 51

1. Object Identifiers (OIDs) 53

B. MANAGEMENT INFORMATION BASE (MIB) 56

1. MIB-2 58

2. Other Uses for MIBs 63

VII. HYPERNODES AND THEIR MIBS 65

A. HYPERNODES 65

1. Network Service Aware Hypernodes 66

a. Extending the Network 68

b. Hypernodes Representing Humans or Groups 70

2. Subnetwork Aware Hypernodes 71

a. Overcoming Bandwidth Constraints 73

b. The Large Network Problem 74

3. Decision Support Aware Hypernodes 78

a. Hypernodes in Edge Organizations 79

VIII. THE TNT-MIO EXPERIMENTS 83

A. TACTICAL NETWORK TOPOLOGY (TNT) 83

1. Mission and Objective 83

2. Experimental Network 84

3. Partners and Sponsors 85

B. MARITIME INTERDICTION OPERATIONS 88

1. What is MIO? 88

2. TNT-MIO 88

3. Collaborative Network 89

4. Hypernodes and TNT-MIO 92

5. A Sample Scenario 93

IX. EXPERIMENTAL METHOD AND RESULTS 95

A. EXPERIMENTAL METHOD 95

1. Groove Virtual Office 95

2. Data Analysis 99

B. ANALYTICAL RESULTS 102

1. Request for Service/Query for Service 102

2. Request for Information (Directed)/(Undirected) 103

3. Response/Amplification 104
4. Unsolicited Information 104
5. Network Registration/Deregistration 105
6. Service Announcement/Status Announcement 105
7. Decision Request 106
8. Decision Announcement/Task Assignment 107
9. SITREP 108

X. CRAFTING MIBS 109
A. A SERVICE AWARE HYPERNODE MIB 110
B. A SUBNETWORK AWARE HYPERNODE MIB 112
C. A DECISION SUPPORT AWARE HYPERNODE MIB 115

XI. EXTENSIONS AND FURTHER RESEARCH 119
A. WEB SERVICES 119
B. SNMP-SNMP GATEWAY 120
C. COMPILED MIBS 120
D. USER AND MACHINE INTERFACES 120
E. TESTING WITHIN THE TNT-MIO DOMAIN 121
F. OTHER DOMAINS/OTHER MIBS 121
G. OTHER DECISION MODELS/DATA MINING 121
H. OTHER CLASSES OF HYPERNODES? 122

APPENDIX A. FORCENET 123

APPENDIX B. RFC-1155 MIB 125

APPENDIX C. FIELD EXPERIMENT 129

APPENDIX D. DEFINITIONS 139

LIST OF REFERENCES 161

INITIAL DISTRIBUTION LIST 165

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

Figure 1. Global Information Grid (from NetOps 100). 5
Figure 2. FORCEnet Concept (McArthur and Mattis, 2006) 11
Figure 3. VIRT Process 21
Figure 4. Five-layer Model 29
Figure 5. 5 Layer Model + 10 31
Figure 6. Organizations and Tasks 33
Figure 7. TMN 35
Figure 8. A Network of SNMP Devices 40
Figure 9. Multi-tier SNMP Architecture 42
Figure 10. SNMP-related OID Tree 56
Figure 11. Line of Sight 68
Figure 12. Loss of Line of Sight 69
Figure 13. Link Restored 69
Figure 14. A Network with Hypernodes. 73
Figure 15. Subnetwork Discovery Flow 78
Figure 16. 802.16 Backbone 85
Figure 17. Experimental Network 86
Figure 18. VPN Sites 87
Figure 19. TNT-MIO Sites 90
Figure 20. Groove Workspace 96
Figure 21. Groove File Sharing 97
Figure 22. Groove Picture Tool 98
Figure 23. Groove Chat Tool 99
Figure 24. Hypernode MIB Tree 110

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

Table 1. FORCEnet Concept 10
Table 2. Layered Models 27
Table 3. TMN Documents (from Subrumanian, 2000) 34
Table 4. SNMP Operations 44
Table 5. MIB/RDBMS Comparison 57
Table 6. proServTable 111
Table 7. childTable 113
Table 8. childMibTable 115
Table 9. madeDecisionTable 116
Table 10. dsInboxTable 118

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I simply could not have done this without the support, encouragement and understanding of my wife. Thank you, Kim.

A good thesis advisor is like a good hairdresser. You never know how it is going to be until it is too late. A good one makes you look good, and once you find a good one, you do not want to let them go. Thank you for everything, Dr. B.

And because the thoughts always sounded better in my head, and sometimes less than spectacular out loud, I would like to thank Mike Clement and all the students in CENETIX for your patience and willingness to let me try out new ideas on you.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

The Global Information Grid (GIG) and FORCEnet, the Navy's 21st Century construct for network-centric warfare, promise a manifold increase in combat power, fueled by information. In many cases, however, we lack the architecture and the mechanisms for the transfer of the requisite information. We present one information architecture, extending the Simple Network Management Protocol (SNMP) that addresses the necessary underlying requirements of such a network-centric transformation and expanding on the previously introduced concept of Hypernodes.

Hypernodes allow for the description of and exchange of information throughout the battlespace. By taking a service-oriented approach to the functions of the military as a network, Hypernodes expose and allow for sharing of these services, connecting providers with consumers and arbitrating data exchange. This is directly in line with the current move toward a service-oriented architecture for the GIG, but extends the concept of services significantly to include human actors, groups, relationships and decision states as important elements of the network.

Further, the utilization of SNMP allows for a common format for exchanged information that is already accessible to most network-attached elements. It further reduces the complexity to implement a service-oriented architecture by removing the need for specialized software. This has the potential of moving closer to the goal of a fully service-oriented GIG by allowing even computing- and bandwidth-constrained elements to participate fully.

Overall the Hypernode construct and SNMP architecture proposed in this thesis suggest an interoperable way to achieve significant impact for network-centric warfare. In this thesis we apply this construct to a campaign of experimentation focused on Maritime Interdiction Operations/Maritime Domain Awareness in order to test its applicability. Utilizing a case-study approach and the constant comparative method of theory generation, candidate SNMP Management Information Bases are developed as a first step toward realizing this architecture.

A. STRUCTURE OF THIS THESIS

Following this introduction, this thesis is divided into 10 chapters. They generally trace this concept from the DoD and Navy-wide visions and problem statements through a formulation of an information architecture based on a service concept to a single operationalization in a contemporary domain of significant interest to the U.S. Navy. Finally, extensions and future research are suggested.

1. Chapter II: The GIG and FORCEnet

The basic premises of the GIG and FORCEnet are discussed, with significant emphasis on developing the concept statements offered by the DoD and the DoN into required actions. The Global Information Grid is explained as a collection of interacting information, processes, infrastructure and human actors in an attempt to clarify and dispel many existing misconceptions. The alignment between the architecture and processes suggested by this thesis and those indicated by GIG and FORCEnet is shown.

2. Chapter III: Push, Pull and VIRT

Network-centric warfare inherently relies upon the ability to transfer information about the battlespace from providers to consumers. There are two leading information architectures for the transfer of such information: information push and information pull. This chapter discusses both information architectures and a developing concept known as Valuable Information at the Right Time (VIRT).

An extension to the existing VIRT framework is proposed, in which intelligent agents, acting on behalf of information consumers and providers, bridge the information push and pull architectures in a manner designed to minimize the impact on either and maximize the functionality provided by core network assets with relatively unlimited bandwidth and computing resources. Finally, the nature of information is briefly covered.

3. Chapter IV: O-I-T and TMN

This chapter offers two viewpoints on the relationships among technology, information and organizations. The Organization, Information and Technology (O-I-T) framework is discussed as the guiding framework for this thesis. O-I-T builds upon commonly recognized multi-layer frameworks, such as the OSI 7-layer model to relate an organization's information, technology and human infrastructures as all three play key roles in GIG, FORCEnet and the Hypernode architecture. The Telecommunications Management Network framework is offered as a contrast to both O-I-T and the SNMP information architecture.

4. Chapter V: SNMP

As the underlying method for information exchange between Hypernodes, the Simple Network Management Protocol (SNMP) is discussed in detail. A basic understanding of the means by which SNMP secures, formats and exchanges information is offered as pretext to the discussion of Hypernodes. SNMP commands are explained in the context of network management, while broader implications of such technology are proposed.

5. Chapter VI: ASN.1 and MIBs

SNMP operates by exchanging information described in various Management Information Bases (MIBs). These MIBs are described in a language provided by the ISO specification for Abstract Syntax Notation One (ASN.1). As Hypernodes will require the development of unique MIBs, this chapter discusses the concepts which are necessary for understanding of such MIB development. A number of well-known MIB variables are used as examples to aid understanding.

6. Chapter VII: Hypernodes and Their MIBs

Three types of Hypernodes, a class of SNMP device beyond managed devices and management stations, are developed in this chapter, expanding on previous work. Network service aware, subnetwork aware and decision support aware Hypernodes are all proposed and explained along with the ways such Hypernodes advance the O-I-T framework and the GIG and FORCEnet concepts. A number of technical limitations are discussed for which Hypernodes offer possible solutions and the specific applicability of Hypernodes to such evolving concepts as Edge organizations is discussed.

7. Chapter VIII: The TNT-MIO Experiments

In order to develop some candidate Hypernode MIBs, a specific campaign of experimentation is investigated. The TNT-MIO experiments, conducted quarterly by the Center for Network Innovation and Experimentation (CENETIX) at the Naval Postgraduate School, provide a campaign of experimentation for which Hypernodes offer much promise. The features of this campaign of experimentation are explained including technologies in use and example scenarios.

8. Chapter IX: Experimental Method and Results

A case study method is discussed in this chapter whereby the communications logs of three prior TNT-MIO experiments are analyzed. After illustrating the technology by which communications took place and were recorded, the results of a thematic analysis are offered. The discovered themes are proposed as a starting point for the development of Hypernode MIBs for the domain under study. Fifteen themes, arranged in 9 groups, are discussed as exemplars of the primary communications among TNT-MIO actors.

9. Chapter X: Crafting MIBs

Three candidate MIBs are developed in ASN.1 to address a number of the themes discovered in Chapter IX. These candidate MIBs represent one of each of the three classes of Hypernodes described in Chapter VII.

10. Chapter XI: Extensions and Future Research

Finally, future research is suggested to build upon this thesis. A number of extensions, including integration with conventional web services are suggested. Next steps, including the development and deployment of the proposed Hypernode MIBs into the TNT-MIO experiment, are suggested. Other areas suitable for analysis are proposed.

II. THE GIG AND FORCEnet

A. THE GLOBAL INFORMATION GRID (GIG)

Underpinning much of the desired technological transformation of the Department of Defense is a requirement for ubiquitous computer connectivity. An espoused vehicle for delivering that ubiquity is the Global Information Grid (GIG). While many people talk about the GIG colloquially, they often use differing and confusing definitions for it. Some of these are, of course, simply incorrect, but most are only incomplete. Since this thesis describes a method for addressing some of the command and control communications requirements for the GIG, it seems helpful to define and discuss it at this point.

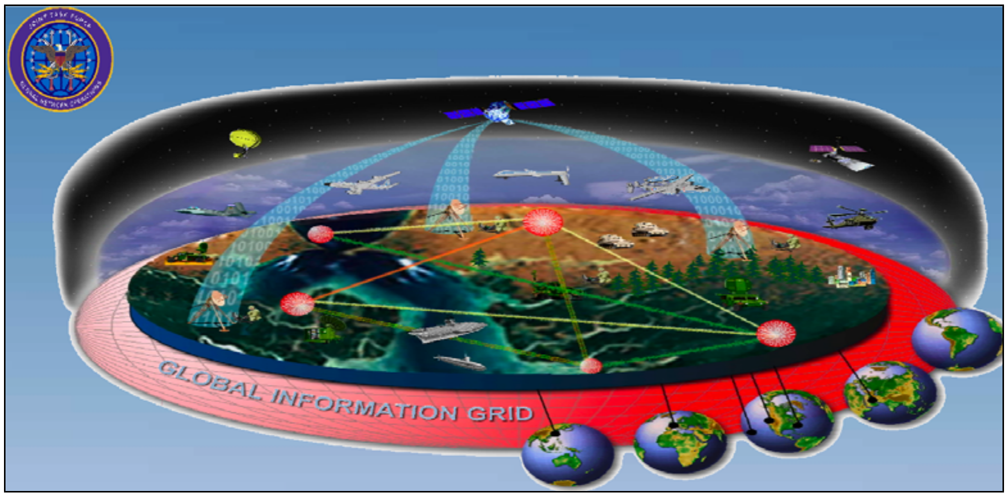


Figure 1. Global Information Grid (from NetOps 100).

1. Definition

According to Department of Defense Instruction 8100.1 , the Global Information Grid is defined as:

> The globally interconnected, end-to-end set of information capabilities, associated processes, and personnel for collecting, processing, storing, disseminating and managing information on demand to warfighters, policy makers, and support personnel. The GIG includes all owned and leased communications and computing systems and services, software (including applications), data, security services, and other associated services necessary to achieve Information Superiority. It also includes National Security

> Systems as defined in section 5142 of the Clinger-Cohen Act of 1996 (reference (b)). The GIG supports all Department of Defense, National Security, and related Intelligence Community missions and functions (strategic, operational, tactical, and business), in war and in peace. The GIG provides capabilities from all operating locations (bases, posts, camps, stations, facilities, mobile platforms, and deployed sites). The GIG provides interfaces to coalition, allied, and non-DoD users and systems.

Further, the GIG "includes any system, equipment, software, or service that meets one or more of the following criteria:

- Transmits information to, receives information from, routes information among, or interchanges information among other equipment, software, and services.
- Provides retention, organization, visualization, information assurance, or disposition of data, information, and/or knowledge received from or transmitted to other equipment, software, and services.
- Processes data or information for use by other equipment, software, or services." (Wolfowitz, 2002)

It might help to take this in pieces to ensure that the reader understands the true nature of the problem space and the environment. First, the GIG is not just technology. Those information capabilities, which are what most people assume constitute the GIG and which are manifest in things like satellite communications, the Defense Information System Network (DISN) or GIG Bandwidth Expansion (GIG-BE), are only one part of the total concept. These information technology (IT) portions are necessary and visible parts of the GIG. That they get much of the publicity associated with the Global Information Grid should not be surprising; they are, however, only foundational.

If the technology is foundational, processes should probably be considered the central construct of the GIG, as it is these processes that actually allow the information in question to be processed, disseminated and stored. Personnel, of course, complete the picture; whether working as network administrators, information providers, data aggregators or simply data analysts, these people are both instrumental in running the GIG and its ultimate customer.

To continue from the definition, there is no demarcation for the GIG between operational warfighting systems and administrative back-office systems. Nor are there any conditions regarding the location of a GIG component. The network-connected dismounted infantryman is on the GIG, as is the DECC (Defense Enterprise Computing Center) at DFAS, as is the LINK-11 connected aircraft in a Carrier Strike Group. The closing bullet points from DoDI 8100.1 fairly clearly spell out that the Global Information Grid is fairly all-encompassing.

In fact, the only non-GIG IT are "Stand-alone, self-contained, or embedded IT that is not and will not be connected to the enterprise network." [Emphasis mine.] Even in 2002, when those words were penned, it was becoming rapidly clear that standalone computing systems were not particularly valuable. Metcalfe's Law suggests that the value of a network increases exponentially as nodes are added. While this seems to be a bit of an overstatement of the value of additional nodes, even pessimistic accountings suggest that the value of a network of n nodes is nlog(n). (Briscoe, et al., 2006) When the potential network is the GIG, we could easily be considering several million computers (NMCI, as an example, is alone serving more than 340,000 desktops), consequently, even where non-GIG IT exists, its value is comparably insignificant.

It is likely appropriate here, for an (very brief) aside about the Navy-Marine Corps Internet (NMCI) and its relationship to the GIG. While NMCI is on and utilizes connectivity provided by the GIG, it is not really “of the GIG.” This is a largely semantic argument, of course, but not an unimportant one. Where the GIG is an all-encompassing construct for interconnectedness, information- and process-sharing, the NMCI is fundamentally just a contract for providing desktop computing services. NMCI neither is, nor aspires to be, a visionary solution to the Navy's Network-Centric Warfare problem. That distinction falls to FORCEnet, which will be discussed below.

2. GIG Services

One thing not mentioned in the definition above is that the processes enabled by the Global Information Grid require a certain number of prerequisite services be available to maximize their utility. At a minimum, in order for information consumers to be able to find data sources, there must be some sort of search and, preferably, a central registry[1]. Additionally, authentication and encryption are very useful features if we do not intend all data sources to be available to all GIG users and wish to protect the data en route (perhaps we need to utilize unsecured channels to send sensitive data).

These services are envisioned as a set of Enterprise Services to be provided be the Defense Information Systems Agency (DISA) and are referred to as Global Information Grid - Enterprise Services (GIG-ES) or Network-Centric Enterprise Services (NCES). While there are 9 currently defined services in this framework: Storage, Messaging, Enterprise Service Management, Discovery, Mediation, Information Assurance, Application Hosting, User Assistant, and Collaboration, the true prerequisites, as I have described them, are Discovery and Information Assurance.

With these two services, an arbitrary GIG-connected user should be able to discover information sources that they require, authenticate to gain access and transmit the information in a secure manner. The architecture proposed in this thesis is designed to interoperate with and relies upon these services. Where solutions are already proposed, such as authentication/encryption via Common Access Card/Public Key Infrastructure (CAC/PKI), they are adopted by this architecture.

The GIG, still, should not be construed as GIG-ES running on some collection of interconnected devices (GIG-BE, etc.)—although these are important and visible parts of the whole. The inclusion of all information processes, capabilities and personnel, irrespective of the nature of the technology (or lack thereof!) upon which they run and outside the strict definition of the 9 GIG-ES services should be obvious and suggests a very broad concept of the GIG. Information is ultimately at the heart of the GIG. If an item for

[1] In the rubric of Service Oriented Architectures, these two are considered together as UDDI (Universal Description, Discovery and Integration). Obviously, UDDI also suggests a few other things like metadata taxonomies (if it is going to be universal description) but those are not strictly necessary as I describe the required services.

discussion transmits, receives, routes, interchanges, retains, organizes, visualizes, assures, processes or disposes of information for the DoD that is part of the GIG; this is true, whether the "item" is a physical technology object, an individual or a process.

B. FORCENET

FORCEnet is all about Command and Control — across the naval enterprise — from warfighting to business practices. It's the way we do business in the 21st Century. We're at the crossroads, the merger of all aspects of FORCEnet. Success will require aligning the systems, the processes, the acquisition, the programmatics and the experimentation needed to bring speed to capability. —VADM James D. McArthur, Commander, Naval Network Warfare Command.

1. Definition

Since its introduction as a part of the "Sea Power 21" concept, FORCEnet has confused and confounded many observers attempting to understand the goals of the enterprise. Many assumed that the “net” in FORCEnet implied a technical architecture for the Naval forces. It seemed to be the logical successor to IT21, and the Naval portion of the Global Information Grid (GIG). While these explanations have a connection to the truth, they are in no way exhaustive of the concept which is FORCEnet.

The original FORCEnet definition described it as "the “glue” that binds together Sea Strike, Sea Shield, and Sea Basing. It is the operational construct and architectural framework for naval warfare in the information age, integrating warriors, sensors, command and control, platforms, and weapons into a networked, distributed combat force." (Clark, 2002) While on the one hand, this is a classic non-definition by the creator of a complicated concept, it also reveals several of the key goals of FORCEnet in very broad, striking terms.

FORCEnet is not only meant to be a technological solution, but also to encompass an operational construct. Further, the technological pieces of FORCEnet are not a single system, or system of systems.; they are not even a single architecture. FORCEnet embodies an entire architectural framework for the information age. It is more accurately construed as a mindset against which all new weapons systems must be gauged. Sensors, shooters and decision makers will be integrated from now on. The “net” in FORCEnet is not just a technological network, but an information network, perhaps even a knowledge network.

Backed by an operational construct, the expectation is that we will leverage increased capabilities in communications and computer networking to link weapons, whatever they are, platforms, wherever they are, and decision-makers, whoever they are, into a seamless, lethal Naval Warfare system. Unfortunately, we had to wait almost 3 years for the publication of a FORCEnet functional concept to expand meaningfully on the sparse definition offered above. (Clark and Hagee, 2005)

2. Functional Concept

When it appeared, the FORCEnet Functional Concept gave us 6 "dimensions" across which development would be required. (Clark and Hagee, 2005) For the purposes of this thesis, these can be subdivided roughly between the original "operational construct" and "architectural framework" as shown in the following Table:

Architectural Framework	Operational Construct
Physical	Cognitive
Information Technology	Organizational
Data	Operating

Table 1. FORCEnet Concept

Clark and Hagee define these dimensions as follows:

- Physical — the various platforms, weapons, sensors and other entities on the operating end of FORCEnet.
- Information technology — the communications and network infrastructure through which these entities interact.
- Data — the common structure and protocols for information handling.
- Cognitive — human judgment and decision making and the human-computer interfaces that support them.

- Organizational — the new force structures and working relationships that will be made possible by FORCEnet.
- Operating — the emergent methods and concepts by which forces and other organizations will accomplish their missions due to the capabilities provided by FORCEnet. (2005)

3. Capabilities

The Naval Network Warfare Command (NETWARCOM) has also released a FORCEnet Capabilities Annex (McArthur and Mattis, 2006) describing the relationship between each of the FORCEnet capabilities and the Joint Capabilities Integration and Development System (JCIDS). Figure 2 attempts to show these relationships. The 15 FORCEnet capabilities can be found in Appendix A.

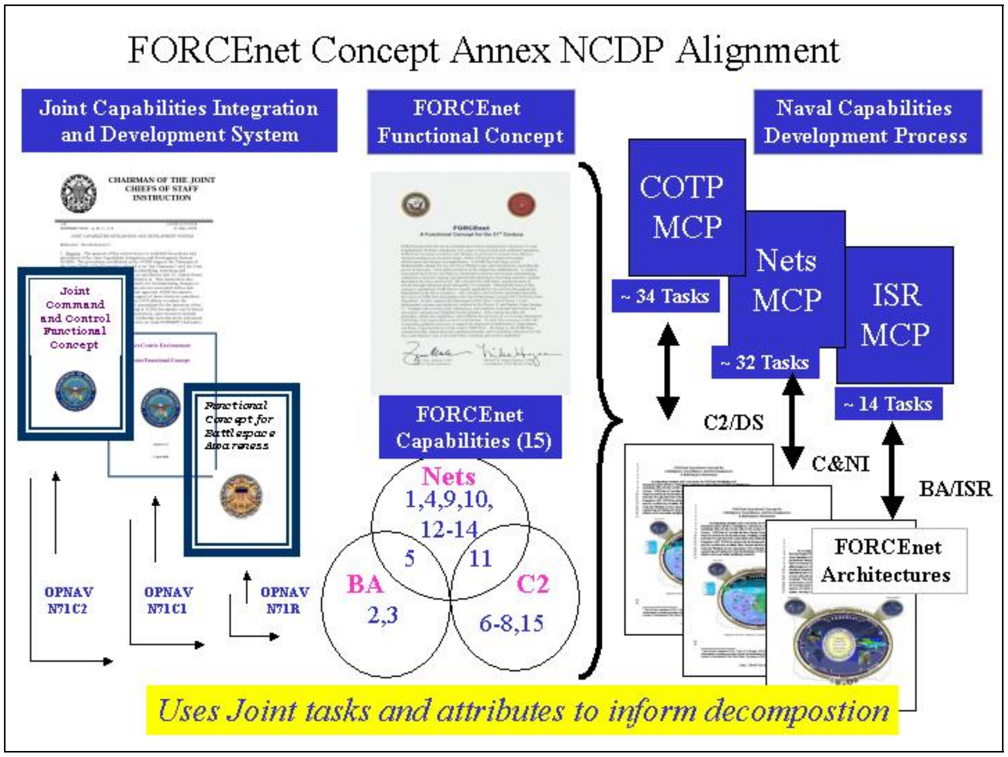

FORCEnet Concept Annex NCDP Alignment
Joint Capabilities Integration and Development System
FORCEnet Functional Concept
Naval Capabilities Development Process
CHAIRMAN OF THE JOINT CHIEFS OF STAFF INSTRUCTION
Joint Command and Control Functional Concept
Functional Concept for Battlespace Awareness
COTP MCP
~ 34 Tasks
Nets MCP
~ 32 Tasks
ISR MCP
~ 14 Tasks
FORCEnet Capabilities (15)
Nets 1,4,9,10, 12-14
5
11
BA 2,3
C2 6-8,15
C2/DS
C&NI
BA/ISR
FORCEnet Architectures
OPNAV N71C2
OPNAV N71C1
OPNAV N71R
Uses Joint tasks and attributes to inform decompostion

Figure 2. FORCEnet Concept (McArthur and Mattis, 2006)

4. Architecture

While the functional concept does not explicitly state it, it does imply that these dimensions are part of a layered architecture, with lower level dimensions (such as Physical—at the bottom of the stack) supporting upper level dimensions, with the Operating dimension at the pinnacle. This thesis adopts a similar proposition, the O-I-T architecture, and proposes that an Simple Network Management (SNMP) architecture of hyper-nodes and decision maker nodes provides a partial solution in the Information Technology and Data domain and direct support of the Cognitive and Operating dimensions.

In the Information Technology dimension, SNMP is a mature and stable protocol for exchanging information among Internet Protocol (IP)-connected hosts. It provides a well-understood platform that is, by design, simple to implement on a wide array of Physical-dimension entities and unencumbered by intellectual property or licensing issues.

In the Data Dimension, SNMP uses a Management Information Base (MIB) architecture for storing and passing data. This architecture allows for the exchange of rich metadata which can be self-describing. This removes the requirement for developing and distributing a comprehensive data dictionary or taxonomy before beginning operations, and allows for the use of new or unforeseen data elements without having to make wholesale changes to unaffected nodes or the underlying architecture. Additionally, SNMP can support public-key encryption, providing for the secure transmission of information over unsecured links and leveraging DoD investments in Public Key Infrastructure (PKI).

This SNMP architecture is, in no way, intended to be a comprehensive solution to the command and control problems of 21st century naval warfare or an omnibus implementation of FORCEnet capabilities. But as Vice Admiral McArthur states in the quotation that opens this section, the concepts of command and control are central to FORCEnet and there exists both a desire and a need for a suite of capabilities to realize the promise of our increasing technological connectedness. Extending SNMP into the Command and Control (C2) realm provides a compatible, flexible and extensible tool in the FORCEnet toolbox.

III. PUSH, PULL AND VIRT

A. ON THE NATURE OF INFORMATION, BRIEFLY

Throughout this thesis, the concepts of information and data will be used repeatedly. It is important to define these in an unambiguous manner since these are pivotal concepts underlying the objectives of the thesis and to reduce confusion for the reader. We borrow directly from the knowledge management literature in this case for our definitions (Nissen, 2006 for a recent and thorough treatment). We consider, then, four different concepts: Signals, Data, Information and Knowledge. While signals and knowledge do not feature heavily in this thesis, they are included for completeness.

Signals are the physical or electromagnetic emanations which are sensed by an agent. Sound waves causing an ear drum to vibrate are signals, as are the voltage levels received on an analog sensor (the electrical output of a microphone is an analog signal—even though it has already been transformed once from a physical signal, as are the electromagnetic variations sensed by radio receivers), or the digital bits sensed across a network link. An important point to understand as we go forward is that signals are the only concept whose instances are actually transmitted or received. This concept will be addressed again later.

Data are some collection of signals which have been codified and expressed in a symbolic manner. Sounds to be processed by the brain are data (we will not here concern ourselves with the neuro/physiological processes by which this happen, it is sufficient to say that the vibrations of the ear drum results in "sounds" to be processed by the brain), as are the recordings of an oscilloscope, or the string of binary or ASCII data read by a computer from the network interface card.

Information is data contextualized. Information has the capability to reduce ambiguity, a feature not found in data. This feature, ambiguity reduction, is consumer specific in that the kind and amount of context required to transform data into information varies depending on the destined recipient(s). Continuing with our examples, when sounds become words in a language we understand they tend toward information. If our oscilloscope is labeled, perhaps in volts and Hertz, and we know where the signal is coming from, that same oscilloscope trace becomes information. Note here that we require some contextual information both about the genesis of the signal and about its representation. A computer file, labeled and formatted as a spreadsheet containing "Fiscal Projections" is also information.

In all these cases, we can now use these contextualized data to reduce ambiguity. We know that someone is speaking, that a certain signal is not another one and that the string of numbers we are looking at is a Fiscal Projection. What if we do not speak the language we are receiving? Are we still receiving information? What if we cannot read an oscilloscope? What if we are not looking for fiscal projections? In all these cases, the signals and data remain the same, but either more contextual information or different knowledge is required to transform our data into information.

Knowledge is also required to turn signals into data. Without knowledge (codified into technology, in both cases) our oscilloscope and our network card have no way of interpreting the signals they are receiving and turning them into data. Similarly, some human knowledge is required to match the sounds we hear against words we know. This transformative capability of knowledge is what makes it special. Knowledge enables action. When we hear the word "Duck!" our knowledge enables us to take action. It is knowledge of how oscilloscope traces correspond to electrical conditions that enable us to fix problems in a circuit and knowledge of business and our economic sector that transforms a fiscal projection into a long-term strategy.

This thesis is primarily concerned with a means for transferring data with enough context to be readily absorbed as information. The transformation from information to knowledge is not specifically addressed, but if we understand our information consumer and their context as well, we may be able to provide them with information better suited for their purposes. Such context could include the origination time of a data message, as well as the entity originating the message. These would seem to be universally useful for generating context. The specific geographical origin of a data message, can similarly be seen to be useful.

With such context, a commander would be able to synthesize or discriminate based on a geographic area of interest or a functional area of responsibility. Perhaps he is the Land Forces commander in a specific area—his context requires information from all land units, regardless of service in that area. Conversely, the senior Army commander may want information pertaining to all his forces, regardless of area or function.

We might as easily conceive of a situation where the commander is concerned with reports only from a certain kind of sensor. Such information would need to include, as part of its context, some details of its origin in order for such a discrimination to take place. This contextual expression and discovery is covered in depth in Chapter VIII. The remainder of this chapter deals with potential architectures for information *delivery*.

B. INFORMATION PUSH

When discussing information delivery options, most of us are familiar with this form of dissemination. In an information push oriented system, information providers determine what information is to be sent and to whom to send it. Push architectures can further be subdivided into "Smart Push" and, although rarely referred to as such, "Dumb Push" systems. A very smart version of such a system is what Hayes-Roth refers to in "Theory 2" (2006).

In a Dumb/Simple/Pure Push scenario, an information provider simply assumes that a consumer will want some piece of information and sends it to them. This is analogous to, for instance, the emails we receive every day from friends and loved ones. Upon finding (or generating) a piece of information, they send it out in an unsolicited manner to whomever they deem appropriate. The information consumer must then determine whether the information is relevant to them or in any way desired.

In a military context, most Defense Message Service (DMS) messages are sent via Dumb Push. The originator chooses which addressees should receive the message as well as what content should be included and pushes it out to them. The list of addressees can (and often does) include wildcard addresses, such as "All Navy Units" or all units within a given region. Dumb Push is rarely an efficient means of disseminating information. There is no guarantee that the receiver of the information actually wished to receive it or that the content or format are such that it can be consumed. It is incumbent upon the information provider to correctly divine an exhaustive list of concerned parties and make the information available to all of them. Worst of all, if information is available from multiple sources, or many information providers are attempting to push information to a single consumer, that consumer can quickly be inundated by a torrent of information that may be duplicative, undesired, incorrect or simply overwhelming in its quantity.

Smart Push, in contrast, relies upon a subscription-based model. An information provider might advertise the availability of a certain class of information—weather data, perhaps. Potential information consumers, then, decide if they would like to receive this information from this source and, if the information provider offers such capabilities, the frequency with which they would like to receive it and the format in which it should be provided. An advanced information provider might also allow for information to be sent if or only if certain triggers are met.

For instance, in our weather data scenario, a unit might choose to receive weather data in a human-readable format every hour or any time it varies from the forecast by more than a specified amount, or any time the wind direction changes by more than 30 degrees, or due to any of a myriad of possible triggers. By choosing information providers and provided information a priori, the information consumer, then, can insulate himself from duplicative, incorrect or unreliable information, and by managing subscriptions is, at least, armed against being overwhelmed by data.

This is, in many ways, precisely the system around which current military Command and Control systems are built. In this archetype, a commander lists his Commander's Critical Information Requirements (CCIRs) and makes them known to his information sources. By doing this, he is effectively subscribing to information streams described by his CCIRs. By crafting them carefully, the commander not only informs his subordinates and other information sources about which information he is interested, he also includes information about the timeliness with which he wishes to receive such information, what priorities are important to him and what should trigger an information push.

While it solves some of the problems of Dumb Push, such as limiting the potential for information overload and reducing unsolicited information, Smart Push, instead, requires information consumers to find their information sources and subscribe to them. Those consumers then must determine which information providers they consider to be reliable and negotiate the finer details of the information exchange. In the simple cases—*send me the local weather observation every hour*, e.g.—this extra overhead is trivial. It is easy to see how this could rapidly become a debilitating amount of work: *From whom should I attempt to get local Intelligence data on this group of people? In what format do I need it? With what periodicity?*

When the problem itself is loosely defined, the amount of work is exacerbated: *I need to be alerted to threats*. Simply formulating the search queries to find the information sources in this case becomes difficult, to say nothing of determining the triggers and formats. Even the sophisticated information consumer is reduced to the proverbial problem of finding a needle in a haystack. Worse, since there are an unknown number of information repositories to be searched and very limited aids to our search, it is more analogous to searching a field of haystacks for a specific bit of hay.

Very sophisticated Smart Push systems can be developed to offset some of these problems. A consolidated search, or UDDI system such as NCES promises might provide us an information provider clearing house at which to begin our searches. As long as information providers are willing to provide rich metadata about their services, it is entirely conceivable that information consumers will find and subscribe to the data sources they require. If this metadata is further consolidated into a single repository (or a number of replicated or federated repositories that appear to the user as a single repository) we can reduce our information field of haystacks to a single stack and arm the searcher with some clues as to both where look and to what the desired information might look like.

As mentioned above, a very special version of Smart Push is described in (Hayes-Roth, 2006). It will be treated below under VIRT.

C. INFORMATION PULL

Complementary to push architectures, there exist pull architectures. This format is also generally recognizable, as it describes the way we interact with web pages and newsgroups on a daily basis. Hayes-Roth (2006) identifies a Smart Pull architecture as "Theory 1." In the most common format, an information provider places information in an accessible location (such as an Internet web page). When new or updated information is required, the consumer must go to this location and access it. There is no guarantee, nor often any indication of whether, the information has changed since the last access.

In a military context, this is the architecture used by a variety of information sources. Navy Knowledge Online, Army Knowledge Online, Collaboration @ Sea's (C@S) K-web and numerous other web-based systems utilize precisely this architecture. When someone posts data to C@S, for instance, all other users of the system will only find the update on their next visit to the site. Information distribution, then, is a function of the speed of updates from our information consumers, as opposed to the speed of updates by the information providers as above. In a military context, it is easy to imagine scenarios where either of these might be preferable, but neither seems to be able to fulfill all contexts.

In our weather scenario above, a pull scenario is equivalent to our user repeatedly accessing the weather homepage to get the current observation. In many cases, the information will not have changed at all since the last report. As observations are made approximately hourly unless there are significant changes, an update frequency of greater then 1/hour will normally generate duplicative information and waste resources. Contrarily, when weather conditions are rapidly changing, which is, of course, when we are most interested in them, such an update frequency may cause us to miss important changes, ultimately changing our plans.

Pull architectures suffer from the same search and metadata problems mentioned above for Smart Push. There is neither a guarantee, nor an expectation, that a user will be made aware of new information sources that become available on the network. Without a well-indexed search capability, or rich metadata for the enterprise-wide discovery, our information consumer finds himself in an even worse situation than above. Here the consumer must first find the data sources and gain access to them. Then, whenever newer information is desired, they must revisit the data sources. When new information is available, there is no notification for the consumer and they may miss any number of updates between information ”pulls.”

D. VIRT

Borrowing terminology from Hayes-Roth and others, we present an extension to their previous work under the rubric of Valued Information at the Right Time (VIRT) (Hayes-Roth 2005, 2006; Oros 2005). According to their theses, the most important aspect of any information architecture is that the decision-maker receives the required information on a timescale appropriate to the decision at hand. The titular "valued" information is that which aids the decision-maker. Further, information in this context is much closer to Shannon's definition of information (1949), in that only the smallest quantum of content that reduces ambiguity is considered information. Additionally, the temporal questions raised above about refresh updates and triggers become central to the discussion of the "right time" to receive said information.

This raises for discussion several points, a number of which have been introduced above:

- How does an information consumer find information?
- How does an information producer decide what information to send? To whom?
- How should data transfers be formatted and transferred such that a minimum of resources are utilized to convey necessary information?
- What metadata is useful?
- How do we handle authorization and security of transfers?
- How do we do all of this while minimizing the impact on the increasingly overloaded decision-maker?

VIRT is one attempt to answer some of these questions. Hayes-Roth compares the pull and push architectures, which he refers to as Theory 1 and Theory 2, deciding that Theory 2 (smart push) is superior to Theory 1 (2006). As he has couched the comparison, it is hard to find fault with his conclusion. What is important to note, however, is that his Theory 2 is only a push architecture from the point of view of the information consumer. It remains a pull architecture form the point of view of the information provider. This hybrid quality, and the use of intelligent agents (termed Condition Monitors and Condition Specifiers by Hayes-Roth) overcomes many of the deficiencies of either model and offloads much of the processing task to systems removed from either producer or consumer.

Figure 3 provides a high-level process view of the VIRT architecture proposed by this thesis. It differs from previous work both in the terminology used and in that we also propose an information architecture for the transfer of information and investigate the specific information used in one experimental case under evaluation. This Figure illustrates the processes by which a single decision-maker receives information from a single information provider for the purposes of simplicity. It should be obvious to the reader that expansion of this concept to multiple decision-makers or to decisions requiring multiple information sources follows the same general procedure.

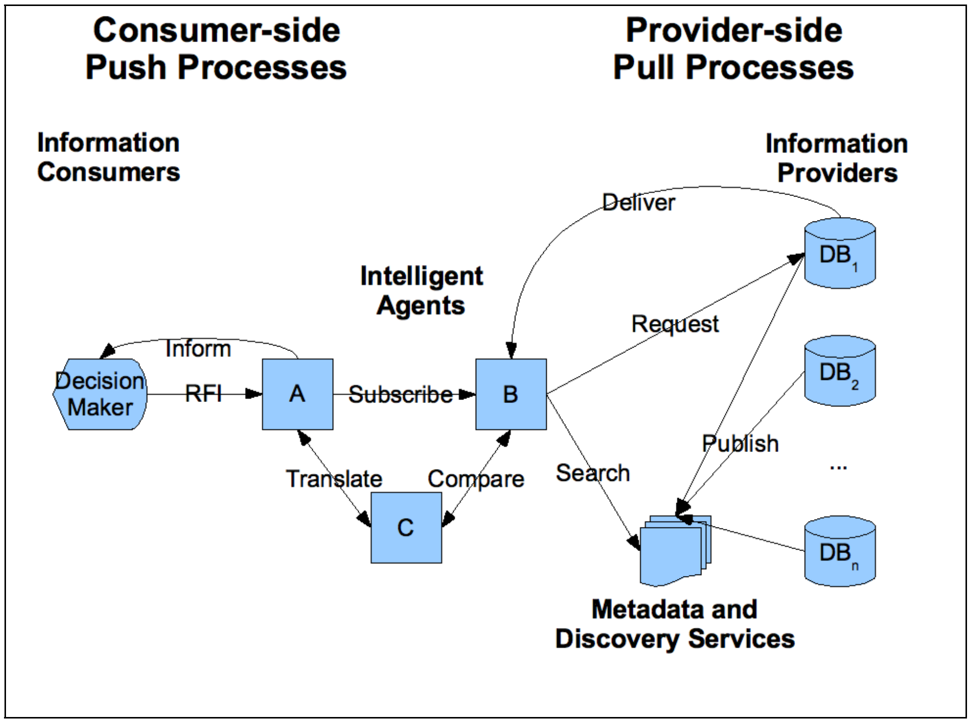


Figure 3. VIRT Process

On the left-hand side of the Figure, our decision maker places a Request for Information (RFI) with an intelligent agent, labeled "A." That agent translates the RFI from a human-understandable to a machine-understandable format and sends the RFI to two other agents. The first is a "subscription" request sent to agent B, while the other is a condition request sent to agent C. While drawn as separate agents, A, B and C represent only separate processes. These may be shared or distributed among computers or networks as required. Some discussion of such discriminating requirements will be addressed later.

The subscription request from A to B also involves a translation of sorts. Where the decision maker and agent A are primarily concerned with information, agent B is only going to be able to search for and deliver data. Consequently, agent A must format the subscription request in terms of the kinds of data needed to fulfill the information request. This is a non-trivial matter, but is within the current state of the art of artificial intelligence for well-understood problems.

A more generic information request to data request translation will be required for this architecture to reach full capability.

At this point, agent B attempts to search for the required data. He draws upon whatever metadata and discovery services are available and known to him, whether via NCES or some other system. After a successful search, agent B then places a request for the required data with the relevant information provider. (Not shown in this diagram are the authentication services to ensure that only permitted users access data and the information security services for en route data.) The information provider delivers the requested data to agent B.

Agent B passes the received data to agent C. At this point, agent B must decide what to do. If the subscription request stipulated a frequency for updates, agent B will wait an appropriate time before initiating another search/request. If other information stores are available for the same data or other data is required, similar requests will now be issued to additional information providers or databases. Agent B, of course, may be responsible for any number of subscriptions at a time from one decision-maker or servicing any number of decision-makers.

Agent C is responsible for comparing the data retrieved with any conditions specified by the decision maker and forwarded in the condition request from agent A. As we discussed above under smart push, perhaps we only wish to know about changes in some data condition from a baseline (i.e. "only alert me if the wind changes by more than 10 degrees or 5 knots from forecast"). In this case, agent C is responsible for filtering out any data which does not meet our criteria. Agents B and C, acting together, fulfill the same capacity as the "Condition Monitor" function described by Hayes-Roth. Separating them, as we have here, allows for better flexibility. The search function in agent B can simply act, while agent C is only responsible for comparing the received data to the specified conditions.

Agent A then reverses the previous translation, returning the machine-understandable data to a human-understandable format and adding the appropriate context to fulfill the original RFI. To our decision maker, this appears to be a "Smart Push" architecture—information is delivered to him when appropriate and without further input from him. From the point of view of the information provider, this appears as a pull architecture—their only responsibility is to post information with appropriate metadata. All of the complexity is handled by a series of intelligent agents which are then responsible only for a small area of interest.

Heretofore we have dealt with an arbitrary agent architecture and, from a generalized viewpoint, it would not be necessary at any point to concern ourselves with a specific implementation. However, for this thesis, we will propose the utilization of the Simple Network Management Protocol as the basis for our information architecture. Both push and pull subsystems are possible with such an architecture, allowing for an implementation very much in keeping with the above Figure, substituting SNMP agents for some of the faculties of the intelligent agents described.

E. BANDWIDTH AND RESOURCE IMPLICATIONS

One of the primary concerns implicit in a transition to Network-Centric Warfare (NCW) involves bandwidth constraints of tactical users. If both information providers and information consumers can be located anywhere in the battlespace, (Clark and Hagee, 2005) we are faced with two choices. We must either provide every network-connected user with enough bandwidth that they are never thereby constrained or we must develop a solution whereby bandwidth ceases to be the driving constraint. As the first is unlikely from a budgetary standpoint, we tend toward the second.

A number of concepts above, such as "Information Producers," were treated in a very abstract manner. Additionally, no mention was made about the location of any of the actors (information providers, agents, information consumers) in the discussion. Both of these points are intentional. The location of the actors in relationship to each other in our architecture is irrelevant. As long as all parties are connected to a network, the GIG network, for example, they are able to communicate. Who or what is consuming or providing information is similarly irrelevant, as is any preexisting command or fiscal relationships among the actors[2].

Because of this, we cannot stipulate that our information providers must possess adequate bandwidth to fulfill some unknown (and arguably unknowable) number of information requests. Nor can we stipulate the equivalent for our information consumer. What we will do, however, is suggest that where available, agents and data repositories should reside on high-throughput links. When publishing data, then, a bandwidth- or resource-constrained information provider should publish it to a proxy location with better resources. Conversely, information consumers will be well-served for their agents to also reside on robust links. This way, the majority of transactions are conveyed over non-resource-limited links.

By adopting such a modular configuration, we allow ourselves the opportunity to house processes wherever they are most appropriate and we remove the requirement for our tactical decision maker, likely on the end of a very constrained data link (32kbps or less in many cases), to make repeated search queries. We also remove the necessity for our information provider, who may also be a tactical unit or a satellite or even an autonomous sensor, to answer repeated queries for information. Below, we will address a specific technology that will allow for this kind of architecture and which supports both push and pull architectures.

[2] Although it is clear that the agents are acting on behalf of the information consumer.

IV. O-I-T AND TMN

A. ORGANIZATION, INFORMATION AND TECHNOLOGY

As a prerequisite for discussing the proposed information architecture of this thesis, we begin by discussing the complex relationships among the military Command and Control Organization, the supporting information technologies and the information itself. Here, we will build upon the concept of a layered architecture as discussed above and adopt the Organization-Information-Technology (O-I-T) model as our point of departure (Gateau, 2006). While only a few of the layers are directly relevant to this thesis, it is important to understand how they all relate, consequently, a review of the O-I-T model follows.

Under consideration is the information system underlying the entire military C2 organization (the GIG, colloquially) as well as the organization itself. The information system will be treated holistically whenever possible, as the data contained within it may be usable in multiple parts of the organization at any time. Conversely, the organization will be investigated at the level of the individual decision-maker, remaining cognizant of emergent capabilities resulting from the grouping of individuals into clusters, and from the network of clusters that form the military C2 network.

1. Layered Models

The use of layered models is prevalent in many disciplines to describe the complex interactions between nodes. In many cases, these layered models are adopted in order to simplify interactions between nodes that do not directly connect to each other, but which rely upon other network members to connect. Most importantly, layered models allow us to abstract away some complex interactions at low levels to focus on interactions at higher levels. This is a manifestation of Wheeler's famous aphorism that "all problems in computer science can be solved by another level of indirection."[3] A synthesis of several models, surveyed below, will be used to develop one appropriate for

[3] David Wheeler, Computer scientist at Cambridge University. This quote is often misattributed to Butler Lampson (Harvard).

understanding the interrelationships between people and technology to enable experimentation and refinement.

Since military C2 is a hybrid network of both humans and information services, some exploration of both is required. A summary of models discussed can be found in Table 1.

In the technology realm, a classic example of a layered model is the 7-layer OSI model used to describe interactions between networked computer and communications devices (1981). In this model, the only connection between devices occurs at the lowest "physical" layer, although information may be relayed between the connected nodes at any layer.

A related model is the DoD networking model (Ennis, et al., 1982), which conveys the same concept in a simpler (4-layer) model. In both cases, the power of the model is actually in the interfaces between the layers. By strictly defining the inputs expected to each successive layer, it is possible to divorce the actions at higher levels from dependency on lower level changes. In this model, for instance, a web application (top layer) does not know or care if it is being sent over Ethernet, wireless, ATM (all lower-level technologies), or any mixture thereof.

Bauer and Patrick seek to extend these models into the human domain (2004). They suggest three layers atop the OSI stack to deal with such ideas as human needs, performance and interfacing, which they term the Human-Computer Interaction (HCI) model. This extension is particularly powerful as it suggests at least one way to deal with the interfaces between the humans and the technology systems with which they interact. Further extensions to cover the sub-organizational, organizational, inter-organizational, etc., levels may allow this model to cover the complete space in which we are interested. Massink and Faconti go even further by extending the OSI model to a continuous interaction paradigm (2002).

Bordetsky and Hayes-Roth (2006), suggest a different sort of addition to the OSI 7-layer model. Their 8th layer allows for adaptive command and control

(C2) of network operations, providing the sort of Business and Service Level feedback suggested by the Telecommunications Management Network (TMN) to the lower levels of the OSI stack in an automatic or semi-automatic fashion. While much of their specific implementation is not directly applicable, their concept of Hypernodes and the application of SNMP in such an innovative way significantly inform this thesis.

Model	Year	Domain	Layers
OSI	1981	IT/Networking	7
DoD	1982	IT/Networking	4
HCI	2004	People and Technology	3 (+7)
Continuous Interaction	2002	People and Technology	5
IW Domains	2001	Information Warfare	3
Knowledge Pyramid	1989	Knowledge Management	3
Von Bertalanffy	1968	Systems	Multiple
Weaver	1949	Communications	3
Kim	1993	Causation	Multiple
Adaptive C2	2006	Network Operations Centers	1 (+7)

Table 2. Layered Models

Alberts, et al., even use a three-layered model to describe three “domains” in which we deal when prosecuting Information Warfare (IW) (2001). In their model, the Physical Domain, Information Domain, and Cognitive Domain exist as separate layers within the IW space. This model begins to grapple with the flow of information, and their treatise on the subject covers a number of “primitives,” such as ”sensing” or “decisions” by showing how data moves up and down between domains as it takes different forms. Unfortunately, their model creates some amount of confusion, in that the physical elements of an information system are not in the physical domain, and that such tools as decision-support aids, which by some argument could belong in the cognitive domain, are relegated to the information domain. This model relates very well, too, to the “Knowledge Pyramid.” (Lucky, 1989 for a good example.)

Some very early research touches, too, upon layered models. Von Bertalanffy, in his discussion of “levels” of organization, and with only a little thought on the relationships between emergent behaviors and those levels, practically invites a layered model for discussing inter-level connections (1968). Even earlier, Weaver, in his appendix to Shannon’s seminal work on communications theory, explicitly describes three levels of the “communications problem” (1949). In this case, he breaks communications into Technical (transmission of symbols), Semantic (interpretation of symbols) and Effectiveness (meaning transfer) levels. His further discussion suggests that symbols are, in fact, all that can be transmitted, and that choice of symbols to be transmitted is the only way to enhance actual communications. Errors, too, introduced at a lower level inherently affect the amount of effective communication that can occur at higher levels.

Somewhere in between, Kim has specifically addressed the sort of inter-level relationships for which the OSI model was originally designed to explain, but in the physical world (1993). He proposes a multi-layer model and spends significant time discussing emergence and how it applies to all systems at all layers.

2. A New Approach

What, then, is missing? Even with the various models describe here, there is no complete model to describe the Decision-Human-Information-IT interrelationships, and this is a prerequisite for appropriately modeling the interactions in a Network Centric Warfare system. Starting from the three “domains” posited by Alberts, et al. seems a fruitful direction for development of an integrated model. A major failing of the Alberts model is that as structured, the three layers do not directly interface with each other—there is something missing at both interfaces.

Between the Physical and Informational domains, resides a Technology layer. Analogously, the realm between Informational and Cognitive domains is populated by Decision Support concerns. It should become clear through further discussion what the requirements are of these layers, and their relationships to other well-know models. To begin, though, it is enough to say that the Technology layer contains those technologies rooted in the physical domain which process, store and manage data and information, while the Decision Support layer consists of systems and processes used to shape information into something usable at the cognitive layer. While the current focus is on IT solutions in these spaces, it can easily be shown that the Technology layer could just as easily represent a manual (human) data collection and management apparatus, and decision support tools “do include decision rules, policy manuals, recollection and staff analysts.” (Huber, 1981) The resulting framework is shown in Figure 4.

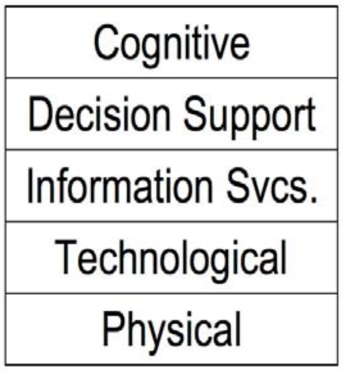


Figure 4. Five-layer Model

While this model is better, it does not fully address the remaining relationships between human and organizational needs, these five layers and the technology stack that will be implemented in support of them. Accordingly, the OSI + HCI model is adopted with some minor modifications as an adjunct to these five layers.

a. The Physical Layer

In the five-layer model, the physical layer is directly analogous to layers 1 and 2 (Physical and Data Link) of the OSI model. In the case of non-computer-networking systems, this physical layer also contains such items as phone lines, hard copy of reports and data (but not the data themselves!) or the actual sound waves in voice communications between people. The physical layer is really most of what we can touch in this model, and as is common with layered models such as this, it is the layer at which all communication ultimately takes place. The physical layer also represents actions and activities that can be sensed or affected occurring in the material world.

b. The Technological Layer

The technological layer contains the systems for moving bits around. It is where, in network terms, the topology becomes logical. It is the counterpart of layers 3 and 4 of the OSI model, describing the Network and Transport mechanisms. In a less computer-oriented view, this is where the file cabinets and folders are as well as routing envelopes and inboxes. The technological layer allows the information and data to move from one node to another within a data system in a meaningful way. While perhaps non-intuitive, automated collection mechanisms reside in this layer as well, such as sensors and systems designed to capture and input data automatically.

c. The Information Services Layer

Although called the Information Services Layer in this model, this layer is the domain of both information and data. In fact, for the purpose of this model, information, data and some types of explicit knowledge can all be interchangeably used. It is the representation, the context or the ability to take action, which is important at this layer, not the content. Here the OSI model fails us, so we will adopt some new language for the sublevels in information. Information in this layer is represented as traffic flows and available services. (Clement, 2006)

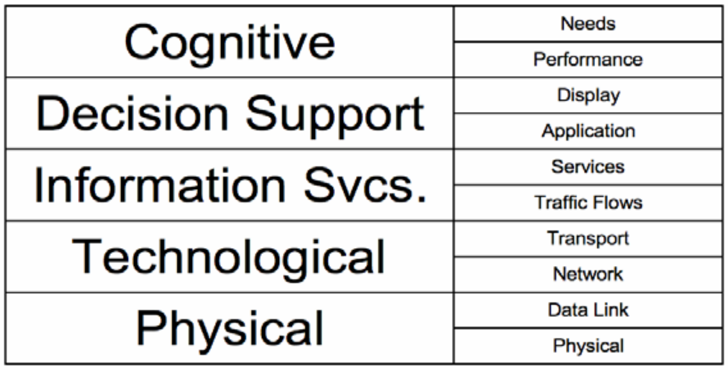
Traffic flows are the bulk information moving within a system. At this level of abstraction, the content of the traffic flows is irrelevant, for instance, management of this level of the model would mean ensuring that the flows arrive at their destination in a timely manner and as prescribed, regardless of, and without a requirement to understand, the content of the flows. Services, however, have some context. They represent the first ability to do things with the information flows, and represent the packaging of traffic flows for use in a service oriented architecture, or existence of “services” in a client-server model.

d. *The Decision Support Layer*

The decision support layer contains the applications with which the user interacts (OSI, layer 7) and the physical man-machine interface (HCI, layer 8). Aside from the physical layer, this is the only layer that contains entities we can touch and manipulate. It also represents the top of the technology stack; everything above this point is about the decision-maker. Applications at this layer are dependent upon the services exposed and the traffic flows within the information layer, but serve to manipulate the raw data, adding context and creating information, or coalescing and enriching information to be presented in a manner suitable for knowledge formation. The display and input devices of the display sub-layer contain the information manifestations to be presented to the human and are the point at which the human in the loop can add other data, metadata and rules to the system.

e. *The Cognitive Layer*

The cognitive layer is the locus of decision-making. "This is the place where perceptions, awareness, understanding, beliefs, and values reside and where, as a result of sensemaking, decisions are made." (Alberts, 2001) Ultimately all information to be acted upon by humans must reach the cognitive layer to processed and synthesized. Human needs and human performance (HCI layers 9 and 10) are also reflected in this layer, as are the existing prejudices, historical and social influences of the decision-maker. The 5-layer model and the associated OSI/HCI layers are shown in Figure 5.



Cognitive	Needs
	Performance
Decision Support	Display
	Application
Information Svcs.	Services
	Traffic Flows
Technological	Transport
	Network
Physical	Data Link
	Physical

Figure 5. 5 Layer Model + 10

f. Task

It is not always customary to explicitly place the task(s) under consideration within the model, but without this explicit mention, it is easy to miss the relationships that exist between task, organization and systems. Additionally, if we are to be prescriptive with our model, it is clear that the type of information systems and the sub-tasks it will be required to perform are heavily dependent upon the primary task or tasks of the organization. Hopefully, this will also cause us to more carefully consider the breadth of tasks confronting an organization in the design of both its information systems and itself in accordance with the task-technology-fit concept (Huber, 1990; Goodhue and Thompson, 1995; Goodhue, 1995).

g. Organizations in the Model

Ultimately, this model must represent organizational actions and interactions with the technology systems, but until now, the organization has not been mentioned as part of the model. As discussed, this model does not actually describe a support mechanism for an organization, much less a command and control organization. What it does describe, however, is the layered model for any information-based decision support system, regardless of whether it supports a single user or an organization and regardless of the tasks to be completed.

3. Organization – Information – Technology

While the HCI model is primarily driven by the interactions of a single user and a computer system, we can easily substitute organizational needs and organizational performance at layers 10 and 9 respectively. Additionally, if the decision support system is taken to be a collaborative space (organizational applications), or providing shared situational awareness (organizational display), layers 7 and 8 also apply across the organization. These substitutions, though, all only reflect the organization as a consumer of the installed technology. In this and many cases, the organization is much more than a consumer that interacts with technology to make a decision. The organization actually is part of the decision-support system, fulfilling roles at the top two layers and controlling information flows and providing services within the Information Services level.

Since the members of the organization can actually transport information (or choose not to), and since they are clearly manifest in the physical domain, it becomes fairly obvious that the organization spans all layers of the model, both as consumers of the information and decision support system as well as part of the system. The complete organizational – informational – technological (O-I-T) model, then, is given in Figure 6.

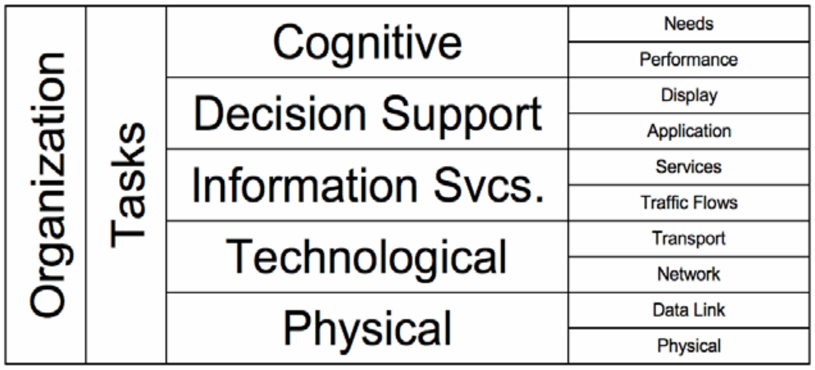


Figure 6. Organizations and Tasks

This thesis discusses the use of the Simple Network Management Protocol (SNMP), an OSI 7th-layer application, with ramifications in the Technological, Information Services and Decision Support layers of the O-I-T model. With very little modification, we will argue for a novel approach to information and decision sharing enabled via this technology. The following section discusses another framework within which SNMP is active. It, too, attempts to relate the entire technology stack, in this case from element management through business management. SNMP will be addressed specifically in the next chapter.

B. TELECOMMUNICATIONS MANAGEMENT NETWORK (TMN)

As a contrasting option, we will now consider the Telecommunications Management Network (TMN) architecture. The TMN also offers a five-layered service model for integrating information technology (IT) management, from the level of individual managed element, with the business processes of an organization. While the TMN originated in the telecommunications industry (hence the name) and was primarily designed to integrate existing proprietary management systems, many of the concepts involved are useful to illustrate a possible alternative to our selected architecture.

Unlike the Simple Network Management Protocol (SNMP) architecture discussed in the next chapter, TMN is dizzyingly complex. It is governed by no less than 12 ITU-T recommendations and is under the purview of 10 different ITU-T study groups. The governing ITU-T document numbers for TMN are shown in Table 3. Three different architecture views are prescribed: Functional, Physical and Information; each describes the telecommunications management network in a different manner, using very different symbology and terminology. Further details about TMN can be found in the documents in Table 3, as the discussion that follows is concerned with only one small subset of the total area treated by TMN.

M.3000	Tutorial Introduction to TMN (1992)
M.3010	Principles for TMN (1992)
M.3020	TMN Interface Specifications Methodology (1992)
M.3100	Generic Network Information Model for TMN (1995)
M.3180	Catalogue of TMN Managed Objects (1992)
M.3200	TMN Management Services Introduction (1996) TMN Management Services 1 (1996) TMN Management Services n (1996)
M.3300	F-interface Management Capabilities (1998)
M.3400	TMN Management Functions (1996)
Q.811	Protocols for the Q Interface - Lower Layer (1996)
Q.812	Protocols for the Q Interface - Upper Layer (1996)
G.773	Protocol Profiles for the Q Interface (1990)
G.774	SDH Network Information Model for TMN (1992-96)

Table 3. TMN Documents (from Subrumanian, 2000)

1. TMN Service Architecture

A fourth, Service, architecture is defined by ITU-T M.3400 and is of much broader applicability that the three previously mentioned. Whereas they are quite specific to the implementation of TMN, the Service Architecture represents an abstract enough account of some of the various services occurring in any IT management to be useful for us. A simplified view of the TMN Service Architecture is give in Figure 7.

Starting at the bottom of the layered model, TMN places the individual **network elements** to be managed. This is an abstract representation of all the various pieces of hardware attached to the network. There is no need for a network element here to only refer to individual physical devices. Any entity that can be individually monitored and managed is a network element in this case. This may be an entire computer system, or may refer to a single port on a managed switch or router. In that case, in fact, the router is itself a network element to be managed (CPU and memory resources, perhaps) in addition to each router port.

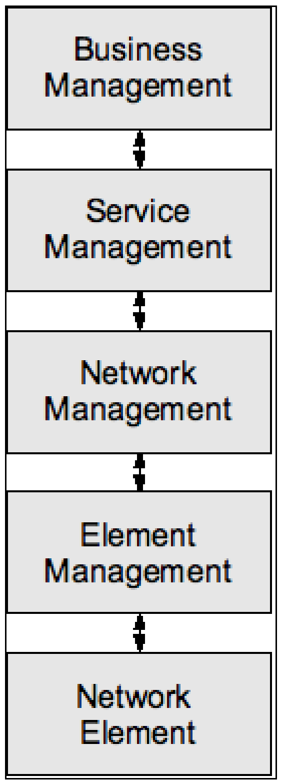


Figure 7. TMN

The **element management** layer of the architecture is concerned, unsurprisingly, with management of the individual elements comprising the layer below. From systems-theoretic viewpoint, we can consider the network elements as atomic. They cannot be further decomposed into subsystems. Here, we manage the properties of the atomic network elements themselves. Element status and configuration are the primary concerns at this level.

When network elements are combined into a network, certain emergent properties arise. The concept of "throughput" for a single network element, for instance, is meaningless. Unless that element is connected to another and transferring data, there is no throughput. When they are connected, and we are interested in throughput, to which element should we ascribe what we measure? In truth, throughput is a measurement belonging to neither element, but emerges from the network created by the two of them. It is at the **network management** layer that we concern ourselves with such concepts as flow control, quality of service, congestion, available bandwidth, etc., that are emergent at this level of the system.

At the **service management** layer we first encounter the concept that there might be a user of our network. Heretofore, we have only been concerned with managing those attributes of the network that exist regardless of who or what may be using it. Service management is concerned with the ability of the network to provide the services (access to date, for example) for which the network was constructed. Additionally, services such as order processing and billing, and trouble ticket management occur at this layer.

Ultimately, the TMN service architecture is concerned with the operation of a business. It presupposes that, as a telecommunications provider, there are certain **business management** tasks that must also be performed. This layer is concerned with such things as human resources and project management. Strategic business decisions (capital planning, infrastructure upgrades) are also involved in this layer of the Architecture.

These architecture layers are obviously interrelated, and a few examples of the interplay among various levels may be helpful to illustrate the complexities involved. Although the terminology in this example is specific to TMN, the concepts are fairly universally applicable. Let us start in the middle, with a trouble call.

When a user reports a service disruption (perhaps she can no longer gain access to her email), a process at the service management layer begins by recording the trouble call and assigning a network technician to the case. Operating, then, at the network management layer, our technician attempts to investigate whether any network anomalies (congestion, dropped packets) could be causing the error. Finding none (or perhaps in parallel) an element management technician begins to check the involved network elements for a failure.

Finding the email server offline, the element manager realizes the switch port to which it is connected has failed. He remedies the situation by selecting a new switch port and closes the trouble ticket (back at the service management layer). Having seen all this, the customer service manager begins to consider options for adding to the network management tools. As this will require a capital outlay, this investigation is in the business management layer. Once implemented, this tool will provide additional capability at the element management and, perhaps, at the network management layers.

TMN and its service architecture model illustrate another method for expressing the interrelationships of the various levels of an organization (in this case, a telecommunications service provider) with its information and technology. In contrast with the O-I-T model, however, there is little consideration for those organizational functions and processes that are not directly concerned with the maintenance and management of the IT network. For this reason, and because of the overall complexity of the TMN architecture for information transfer, we will adopt a different architecture, namely SNMP.

THIS PAGE INTENTIONALLY LEFT BLANK

V. SNMP

A. HISTORY AND ARCHITECTURE

Originally defined in the Internet Engineering Task Force (IETF) Request for Comments (RFC) 1067, the Simple Network Management Protocol (SNMP), is remarkably well-named. It is primarily used for the management of network-connected devices and is brutally simple. The original RFC 1067 implementation of SNMP, referred to post-hoc as SNMP version 1 (or SNMPv1) was modified several times and is currently specified by RFC 1157. It was further amended by RFCs 3416, 3417 and 3418 into a version known as SNMPv2. A third incarnation (SNMPv3) is now a full standard and brings significant security additions, including Public Key Infrastructure (PKI) to the protocol.

Although SNMPv1 dates from 1990, and is now technically a historical standard, it is the most widely implemented version. SNMPv1 software is available for a dazzling array of devices. Every modern operating system implements SNMP (at least to v1 although most support v2 as well), as do most routers, managed switches, wireless access points, (network attached) printers, some network cameras and even the odd toaster.[4]

SNMP is a client-server architecture consisting of Managed Devices and Management Stations. The managed devices on a network run the server side of the SNMP software, known as an SNMP agent, responding to commands issued by the management station(s). These managed devices, classically, are hubs, routers, switches, servers, PCs, printers or any other network-connected devices. The management station is usually a workstation or server with significant resources, utilizing a database and sophisticated software to retain and display the current and historical information provided by the managed devices. A simple network of SNMP-connected devices is shown in Figure 8.

[4] Famously, John Romkey, then an employee at Epilogue software once demonstrated an Internet-connected toaster (a Sunbeam Radiant Automatic) at the 1990 Interop conference. The toaster ran TCP/IP and was controlled by SNMP! (Living Ineternet, 2006; Romkey, 1995)

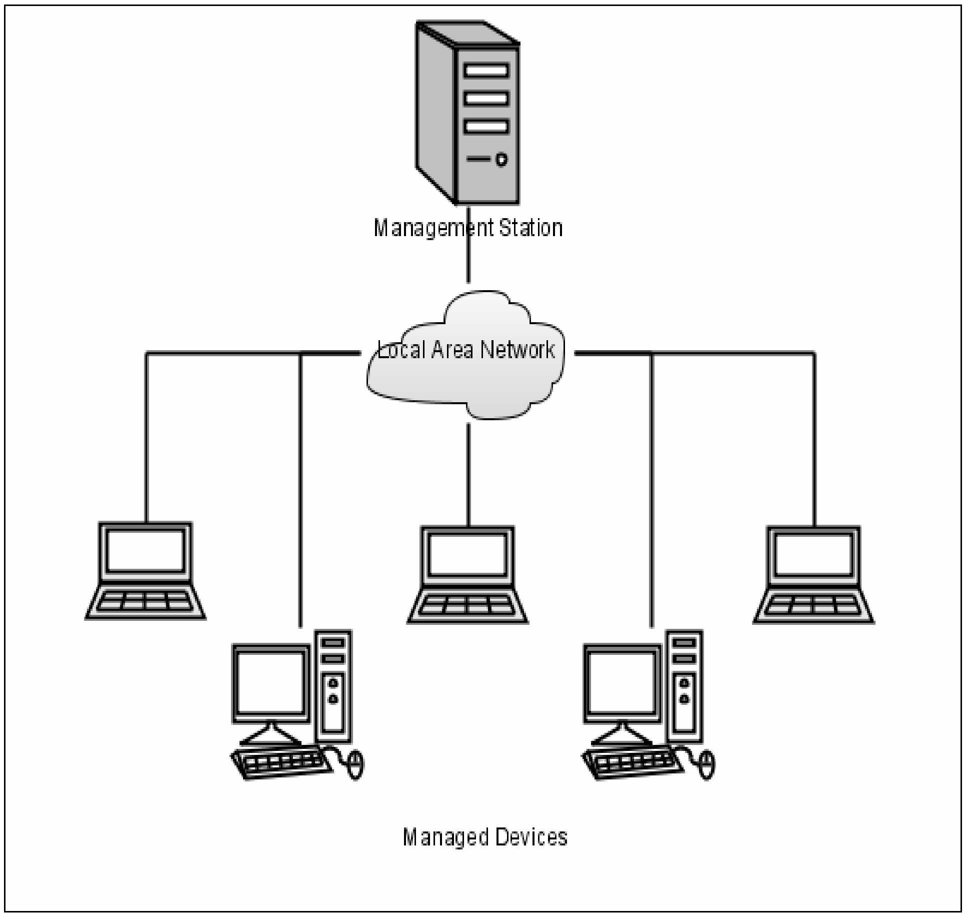


Figure 8. A Network of SNMP Devices

This illustrates a construct inverse to the normal concept of a client-server architecture. Here the management station runs the client software, but is likely one of the more powerful machines on the network. The managed devices, although they run the server portion of the SNMP software, may be very simple devices, and may not even provide any other means of monitoring or management. Due to potential for confusion, then, we will attempt to consistently use the terms *managed devices* and *management stations* to refer to the network nodes themselves and *SNMP agent* and *network management system (NMS)* to refer to the server and client software pieces to help alleviate confusion.

Unfortunately, that confusion is likely to persist due to the fact that every management station is potentially a managed device, and some managed devices may also be management stations. At the very least, management stations normally also run the agent software so that they can report on their own status and include that in the overall network picture. Also, several commercially available network management systems that rely upon SNMP allow for a multi-tiered implementation (see Figure 9), meaning that whether a device is managed or a management station is largely a matter of perspective. This is analogous to the way in which one level of an organizational hierarchy consists of either bosses or subordinates depending upon where the interested party sits in that same hierarchy.

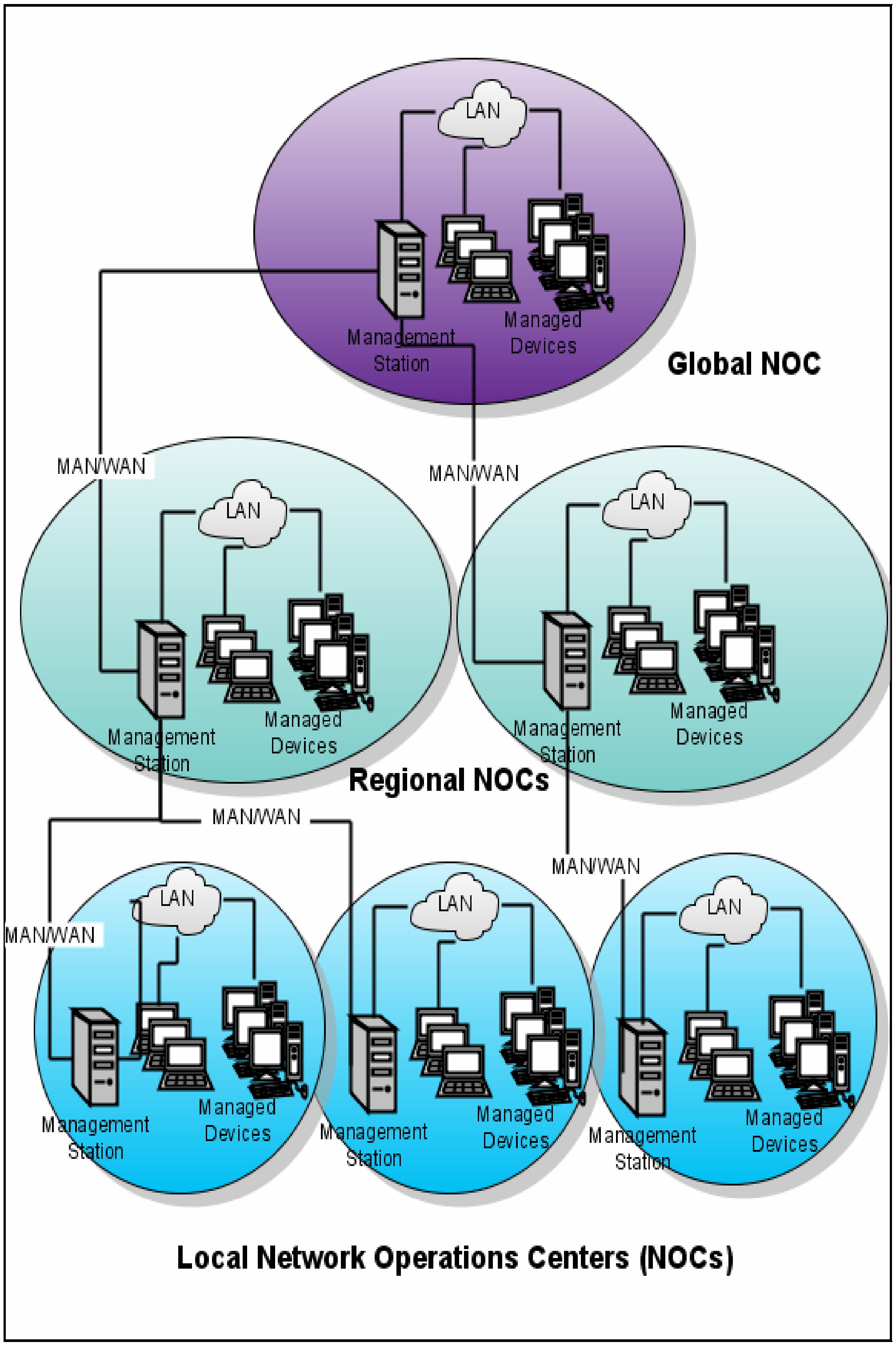


Figure 9. Multi-tier SNMP Architecture

In Figure 9, each local area network is managed by a management station within their Local Area Network (LAN). In this way a local Network Operations Center (NOC) is responsible for the Fault, Configuration, Accounting, Performance and Security (FCAPS) management of their local devices. Additionally, aggregate information and/or specific outages and statuses are forwarded up to superior NOCs. The green ovals in the center of the Figure represent regional NOCs. Each of these is, again, responsible for management of the nodes directly attached to their LAN as well as being responsible for monitoring the statuses of their subordinate NOCs passed to them across Metropolitan or Wide Area Network (MAN or WAN) links.

The purple global NOC at the top of the Figure is ultimately responsible for managing the entire global network (in our instance, that global network is the GIG). It should be fairly obvious that simply forwarding every status up the chain will put an unnecessary load on the management systems at higher levels of the hierarchy and that the global NOC is unlikely to be concerned about every outage on a switch or router on every LAN in the enterprise. Therefore, most systems are configurable to only report significant changes in status to higher levels.

Also, as Mauro and Schmidt suggest, this tiered system could allow for local NOCs to be staffed only during working hours, relying on the regional NOC to monitor the network when no local staff is present (2005). When local staff is present, few alerts are forwarded to the regional NOC and local technicians monitor and maintain the network. If, for instance, that local LAN is only utilized in a 8 hour/5 day manner, 24 hour/7 day monitoring may be infeasible or unnecessary. After hours, the local NMS could be set to forward outages to the regional NOC who may then choose to recall local personnel to repair the system, attempt to fix it remotely or simply decide the outage can wait until the local personnel return to work.

B. POLLING AND ALERTING

SNMP's simplicity derives, in part, from its reduced instruction set. In SNMPv1, the command set is limited to three: `get`, `set` and `trap`. A fourth command, `getnext`, is also included in the specification, but it is merely a specific and limited form of the `get` command. All management information, then, is made available and transferred or modified by these three commands. The `get` command is used by management stations to retrieve status information from a managed device, the SNMP agent responds with a `getresponse` message. `Set` is used to modify a parameter on a managed device and `trap` is used by a managed device to send an unsolicited message to a management station. Table 4 contains a complete set of SNMP operations and the versions which support each. Additional operations were added in SNMP versions 2 and 3.

Name	Minimum SNMP Version
`get`	1
`getnext`	1
`getbulk`	2
`set`	1
`getresponse`	1
`trap`	1
`notification`	2
`inform`	2
`report`	3

Table 4. SNMP Operations

Ultimately, all these operations amount to information transfers of the types discussed in Chapter III. The polling methodology used by the get commands is directly analogous to the operations described by a Pull architecture. Similarly, the trap, notify and inform operations enable information transfer according to a Push architecture. In both cases, it should be noted, SNMP only handles the information request/delivery portion of the operation. Search, especially, and authentication require processes outside those which SNMP provides.

1. The `get` Family of Operations

As they all share a procedural familiarity, `get`, `getnext`, `getbulk`, `set`, and `getresponse` will be treated as a family of operations. `get` is the patriarch of this family. Utilizing this SNMP command, an NMS asks a single device to return the value of a single data element. While we will cover the details of these data elements later, the `get` command is not unlike a directed query to a database, seeking, for instance, the phone number of a single client. Both the network address of the target device must be known (as well as any security credentials required) along with the identity of specific data element under consideration.

Upon receiving a `get` request, the SNMP agent on the target device issues a `getresponse` to the requesting management station. The `getresponse` operation returns the requested value to the management station, utilizing a User Datagram Protocol (UDP) packet largely identical to the incoming request; only the Protocol Data Unit (PDU) type has changed from 0 (get) to 2 (response). Further details about the packet-level details of SNMP, for those interested, can be found in Mauro and Schmidt (2005).

The `getnext` command operates in much the same was as the `get` command. A management station issues the command to a single managed device for the value of a single data element. In this case, however, the NMS does not need to know the identity of the specific data element requested. In fact, the `getnext` operation simply asks for the next value on the agent. Following our previous example, then, if we know that there may be more data fields after the phone number of our client, we could continue to issue `getnext` commands until the database had exhausted fields. Each one would reference the previous data element's address (which is, of course, now known to us), starting with the phone number and ask for the value in the next field.

`Getnext`, then, allows us to "walk" the entire database on the agent by simply continuing to retrieve the next value until all have been retrieved. Unfortunately, this, like get, requires issuing a separate command to retrieve each data value. In SNMPv2, the developers decided this was terribly inefficient and introduced the `getbulk` command. Unlike `get` and `getnext`, with `getbulk`, an NMS can request multiple data values to be returned in a single transmission. Like `get`, `getbulk` requires prior knowledge of the identity of the data elements to be returned.

In contrast to the `get` commands, the `set` command is not utilized to retrieve data, but rather to input data. A `set` operation is identical to a `get` operation, except that in addition to the identity of the data element to be modified, the NMS sends a value. That value is received by the agent and, permissions allowing, stored in the data field specified. While often not heavily utilized, the set command allows SNMP to be used as an active management protocol, making changes to managed device configurations, as opposed to a purely passive monitoring protocol. Using the `set` command allows one, for instance, to enable or disable a network interface or change the IP address of a remote managed device.

a. Security Concerns

The power of the `set` command comes with serious security concerns, however. Since this operation can be used to disable a network interface or change network configurations, it is a possible vector for attack. Unfortunately, version 1 of the SNMP specification includes only the most rudimentary of security capabilities. Two community strings are defined for each agent, a read-only string and a read-write string. Anyone with these community strings (which default to "public" and "private," respectively, on most SNMP implementations), can discover many configuration details about network-connected agents and can make significant changes to the network, disrupting or denying service and generally interfering with network operations.

SNMPv2 increases security in a very modest way, by allowing an arbitrary number of community strings. Each community string, then, allows specific access rights to specific variables. Each agent may allow access to each community string irrespective of how other devices are configured. We might, then have a community called "SeniorAdmins" which allows read-write access to all variables on all devices and another called "DCAdmins" which allows read-write access, but which is only enabled on the machines at the Washington, DC location. Those with the DCAdmin community string would only be able to modify the machines in DC.

Unfortunately, there is no magic to these community strings. Community strings are rather like username/password pairs, but with both items confused.[5] Simply possessing a community string is sufficient to utilize it, as there is no separate challenge or pass phrase required. If a naming scheme like the above is used, how hard would it be to guess the read-write community string for a Norfolk location? Anyone who had ever had the SeniorAdmin community string would forever be able to modify any device on the network.[6] To make matters worse, these community strings are passed in cleartext as part of the SNMP operation. Ultimately, anyone capable of intercepting SNMP messages on the network would be able to gain access to any communities in use.

These security concerns have significantly hampered the adoption of SNMP in a read-write capability, and many enterprises, including the Department of Defense recommend disabling SNMP on all devices (DISA, 2006). All of these security concerns are addressed by SNMPv3, its adoption of user-based security that can utilize PKI, and the fact that all credentials (and potentially all datagrams) are sent encrypted. SNMPv3, however has never garnered the market penetration that characterizes SNMPv1 (and to a lesser extent v2). This may be due to the increased complexity of the implementation or may simply result from a supply/demand Catch-22. Since active SNMP management never caught on due to the security concerns few people are

[5] Here we use a slightly disused form of "confused" meaning "mixed up." We specifically mean that the username and password pair are mixed up together into one item, or "fused" together (con-).

[6] The only realistic way to fix this problem, then, is to change the community string on every affected device in the network any time we need to revoke access. In many cases, this can mean making a change to every single managed device on the network.

demanding agents with the v3 security features. Additionally, by using other security options (such as SSL tunneling) v1 and v2 implementations can be significantly hardened.

2. The `trap` Family of Commands

Like the `get` operations above, `trap`, `inform` and `notify` also share a family resemblance. They allow an SNMP agent to send an unsolicited message to a management station. In this way, push-type messages can be sent, adding to polling an alerting capability for the SNMP architecture. Both of these will be utilized for information dissemination by the solution proposed by this thesis.

There are obvious network management-related reasons for using such messages. Since polling every device in the network is potentially expensive in terms of network resources, it is customary to increase the polling interval as much as practicable. Unfortunately, this means that significant changes to network status can occur undetected for a fairly long time. Utilizing `trap` operations means that critical messages can be sent to the NMS between polling intervals. When a previously designated condition is met, such as an interface falling off line, the SNMP agent issues a `trap` message to a similarly previously designated management station informing it of the change.

The `trap` operation is also restricted to sending a single data element/value pair per message. These data elements are partially defined by the SNMP specification, but additional data elements may be defined by the user for use in their enterprise as trap messages. Consequently, the user can choose to send trap messages for whatever change in conditions they deem appropriate. Additionally, multiple `trap`s can be sent based on the same condition to multiple NMSs or containing different information.

SNMPv2 introduced a `notify` operation to supplement `trap`. Part of the reason for this is purely technical. `Trap` messages follow a different form than do `get`-style messages; `notify` utilizes the same form as `get`, but with a different PDU type (this time type 7). The other difference is more significant. Whereas `trap` is limited to a single data element/value pair per message, a single `notify` message may include any number of data elements and values. Neither `trap` nor `notify` messages are acknowledged by the NMS to which they are addressed. This means that an agent has no way of knowing that a message got through. This is by design, of course, since `traps` are often issued for network problems resulting in a condition where the message can never get through. Requiring acknowledgment in such a scenario is unwieldy.

To account for situations where an acknowledgment is necessary, SNMPv2 specifies an `inform` operation. The implementation of `inform` is identical to that of `notify`, with the exception of the acknowledgment. While not specifically recommended for network management operations, the kind of applications suggested by this thesis will make heavy use of an acknowledged push-type message, heavily favoring the use of `inform`..

3. The `report` Command

The remaining push-type operation is `report`. The `report` operation was specified in SNMPv2, but not implemented until v3. The `report` operation is utilized to allow SNMP engines to communicate with one another, for instance, for passing error messages about malformed SNMP messages.

C. SNMP'S BROADER APPLICABILITY

The simplicity of SNMP's command set allows for significant flexibility. As we will see in the next chapter, the structure of management data in SNMP is wholly agnostic about the content of the data. There is no reason that the SNMP agent needs to pass only network management data with a `getresponse` or a `notify` message. In fact, the agent need not be concerned at all with the meaning of the data element/value pairs it will pass.

Similarly, although most SNMP client applications are specifically tailored for handling network management information, there is no restriction that they must do so. As long as they have some knowledge of the data structure about the agents for which they are responsible, *any kind* of data can be successfully transferred, often with a significant amount of context. It is this flexibility of SNMP that we propose to exploit for the transfer of C2 information in an automated fashion.

VI. ASN.1 AND MIBS

The power and flexibility of SNMP stems not only from the simplicity of its command set, but also from the inherent flexibility offered it by the underlying language and data structure. The structure of management information in SNMP is described in a notation known as Abstract Syntax Notation One (ASN.1). ASN.1 allows for semantic definition of SNMP data while remaining agnostic about the type of transfer syntax used or the specifics of the lower-level (OSI) protocols in use beneath SNMP.

The actual data elements within an SNMP managed device are described in ASN.1 and collected as something called a Management Information Base (MIB). Management Information Bases are best understood as database schemas. They do not, themselves, hold any data, they merely describe the data that is to be maintained by the SNMP agent and contain necessary metadata. This metadata (expressed as ASN.1 definitions) is the key to the exchange of SNMP data.

MIBs exist for a large number of purposes, primarily for network management. Using ASN.1, however, it is possible to construct a MIB for any arbitrary data desired. By leveraging this inherent flexibility of ASN.1 and MIBs, we will describe a method for extending SNMP beyond network management.

A. ABSTRACT SYTAX NOTATION 1 (ASN.1)

If SNMP were being designed today, it is unlikely that the designers would choose to use ASN.1 as the lingua franca. Some form of Extensible Markup Language (XML) would likely have been adopted for the description of data elements. However, SNMP predates XML significantly, and the heavy influence of CCITT and ISO in the development of SNMP made it an obvious choice (ASN.1 is an ISO/ITU-T joint standard). Fortunately, ASN.1 is sufficiently extensible and self-documenting to suit our purposes. For details on the ASN.1 standard beyond the scope of this thesis, see (ITU-T Rec. X.680 (2002) | ISO/IEC 8824-1:2002).

Like XML, ASN.1 allows the encoding of a large number of data types; unlike XML, ASN.1 predefines the format of such data types, much like a programming language. For instance, there is a BOOLEAN data type which allows only for true or false values. Similarly, there are CharacterString (a set of characters) and REAL (a real number, in scientific notation) types defined in the "universal class." Individual data elements are assigned to one of these types or a structured set of them using a formal syntax of the Backus-Nauer Form, which looks like:

```
<name> ::= <definition>
```

(Subramanian, 2000)

Here, both <name> and <definition> are entities and ::= is read as "defined as." For instance, if we are interested in defining the status of a computer, we might choose to use:

```
IsUp ::= BOOLEAN
```

Then, the variable "IsUp" could take the value of either true or false, signifying the possible states of our system. A more complicated example, where the system might be either up, down or in some sort of error state could take the form of an enumerated list, such as:

```
SystemState ::= ENUMERATED
     {
          down(0)
          up(1)
          error(2)
     }
```

The enumerated list could, of course, be expanded to include any number of variables identifying different system states. The numerical value (in this case 0, 1 or 2) is the information held in the variable, the text information (down, up, error) is only maintained as part of the definition.

The definitions in ASN.1 may also be nested, building complex classes of variables. Let us take the following example for discussion:

```
          System ::= SET
               {    SystemName     CharacterString,
                    Location       CharacterString,
                    SystemState }
```

Now, our system is defined by a set of variables. There are two character strings for the name and location of the system, and a third variable, SystemState. As long as our definition of SystemState (from above) appears in the same ASN.1 document, this is a valid definition. Likewise, instead of the freeform character string for location, we could have defined another enumerated list of the valid locations for our systems and used that. It should be fairly obvious that very complex constructs can be created using ASN.1.

Above, we mentioned that any definitions within an ASN.1 document could be used within later (or earlier!) definitions in the same document. By using a special keyword, IMPORTS, we can also invoke definitions from other ASN.1 documents. This is analogous to using the #include declaration in the C programming languages. By the same analogy, we can simplify any specific ASN.1 document to the extent that we can draw upon these external references. Similarly, we can enforce a level of standardization among our documents to the same extent that they share references.

SNMP has, to a large extent, done this by creating several high-level ASN.1 documents defining commonly used network management variables, classes and constructs. The Internet Engineering Task Force (IETF) Request for Comments (RFC) 1155 (included in Appendix B) and RFC 1213 (McLoghrie and Rose, 1990, 1991) include a very large number of common definitions. These include both low-level object definitions (such as IP addresses) and high-level aggregations of objects (such as the table describing a network interface: ifTable).

1. Object Identifiers (OIDs)

While there is neither the space nor inclination in this thesis to cover ASN.1 in depth, there is one other special kind of ASN.1 object that is of specific interest to us. The OBJECT IDENTIFIER (OID) is a numerical representation of each occurrence of an object. This allows us to do several things in a fairly simple way. We have a single numerical reference that can be used to return any value encoded in our document. Instead of referring to the textual name of the object, we can simply use the OID.

Additionally, by using OIDs, we can disambiguate between multiple occurrences of a data object. For instance, an SNMP-managed system might have two Ethernet Network Interface Cards (NICs). The ASN.1 object describing interface status (up, down) needs to exist for each of these NICs separately, as they can change and may need to be accessed individually. ASN.1 allows for such unique enumeration of duplicate objects.

OIDs are arranged in a tree format and expressed in dotted decimal format such that 1.3.6 is a child node of 1.3 and a parent node of 1.3.6.1. OIDs comprised of only a single number are children of the "root" node. Although there is no explicit OID for the root node, it is often suggested by expressing OIDs with a leading decimal point. Consequently, 1.3.6 and .1.3.6 are equivalent forms of the same OID, with the second form including an explicit reference to the (unwritten) root node. All OID references in this thesis will utilize the first form without the leading decimal.

OIDs which have no children are referred to as "leaf nodes." These leaf nodes are appended with a trailing 0 if there is one and only one value possible for that data object. For instance, a single computer can have only one system location. The OID for system location is 1.3.6.1.2.1.1.6. The value assigned to a specific computer's system location is stored with an OID of 1.3.6.1.2.1.1.6.0. The OID for interface status (addressed generically above) is 1.3.6.1.2.1.2.2.1.8. Since each interface can have a separate status, an additional number is appended to this for each unique interface. This number is used uniformly for reference, such that if it is interface number 24 (perhaps the device is a large switch) the interface status would be held in 1.3.6.1.2.1.2.2.1.8.24 and other interface-specific information (such as the IP address) would also end in .24.[7]

The first-level branches off the root node number only three and are assigned as: 0-ITU, 1-ISO and 2-ISO/ITU (joint). For SNMP, the OIDs of interest are primarily located in a portion of the tree defined as:

```
     internet OBJECT IDENTIFIER ::= {iso(1) org(3) dod(6)
internet(1)}
```

this is equivalent to 1.3.6.1, and according to the rules of ASN.1 syntax it could have equivalently been written as:

```
     internet OBJECT IDENTIFIER ::= {iso(1) org(3) dod(6)
(1)}
```

since the final "internet" is assumed from the name of the OID, or as:

```
     internet OBJECT IDENTIFIER ::= {(1) (3) dod (1)}
```

In order to correctly parse this third form, we would obviously have had to define the "dod" OID elsewhere. Additionally, it should be fairly obvious, that although all three forms are acceptable and equivalent, the first form is more easily human readable and should be preferred in most cases.

A graphical representation of the segment of the OID tree most applicable to SNMP is shown in Figure 10. A web-based OID browser is available on line at http://www.alvestrand.no/objectid/top.html which enables users to navigate the entire OID space.

[7] With the exception of the last one, each example so far has utilized OIDs as a string of decimal dotted single digits: 1.3.6.1, etc. There is no realistic limit, however, to the width of each portion of the OID tree, and multiple-digit OIDs are completely valid. For instance, the USA is granted a section of the OID tree under 1.2.840 and public organizations beneath that start at 113533, meaning that the organizational OID reference for MIT, for instance, is 1.2.840.113554. The single-digit sort of OIDs are used only because they are germane to SNMP and because they are easier to write.

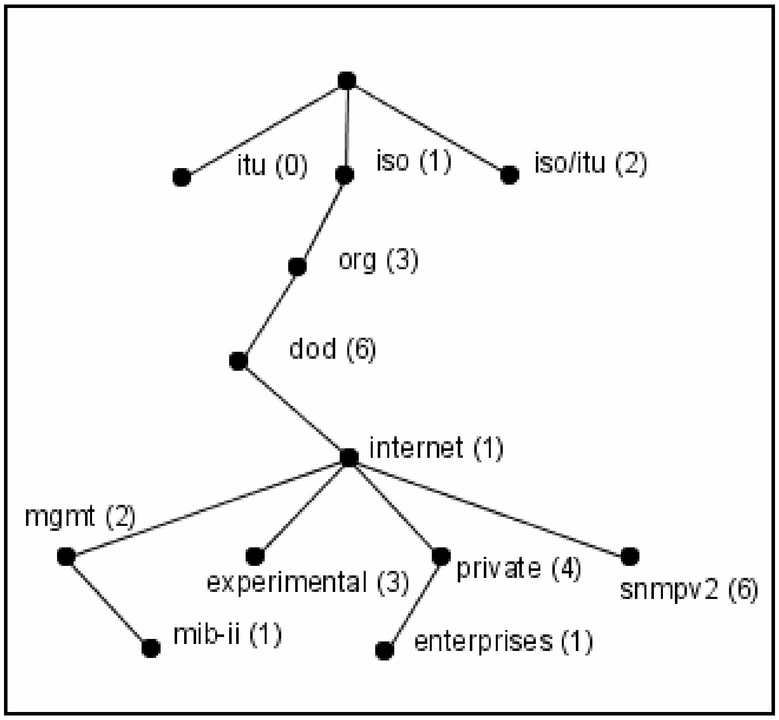


Figure 10. SNMP-related OID Tree

At the top of Figure 10, you will notice an unlabeled node. That is the root node of the OID tree. Not shown, of course are a very large number of child nodes for each of the displayed nodes, as only those generally relevant to SNMP are shown. At the bottom of the tree there is a node labeled "mib-ii." This node, known as MIB-2, and bearing the OID 1.3.6.1.2.1 is the parent for most of the general SNMP management information. Information specific to particular network devices or vendors is located beneath the "enterprises" node 1.3.6.1.4.1. ASN.1 documents describing SNMP management information are commonly referred to as Management Information Bases (MIBs).

B. MANAGEMENT INFORMATION BASE (MIB)

As mentioned above, MIBs can be best understood as database schemas, written in ASN.1 and describing the SNMP management information apropos to a given system or device. The use of the term "base" in the name as well as the relationship between network management systems and databases of network management data tend to confuse casual readers into assuming that the MIB is a database on an SNMP device which contains the management data. This is both understandable and regrettable. There need be no relationship between a MIB and any database and certainly none is required. A Table describing some of these differences follows:

	MIB	RDBMS[8]
Provides a *format* for describing Data?	Yes	Yes
Includes metadata?	Explicitly	Only when/if desired
Entity relationships?	Hierarchical (by OID)	Unconstrained
Provides data storage?	No	Yes
Human-readable definition?	Yes	Rarely
Allows sophisticated queries?	No[9]	Yes
Requires additional software?	Generally, No	Yes

Table 5. MIB/RDBMS Comparison

What is true, however, is that the MIB describes the SNMP data accessible by an agent and the relationships among those data. Each agent can decide how it wishes to actually store and handle the management data. Some systems may actually utilize a database (relational or a flat file), others may generate the information as requested, returning, for instance, the uptime of a system as the result of locally executing the Unix “uptime” command and reformatting the information. Most systems will use some combination of both. The MIB, rather than being the location of such data, is merely a translation and description mechanism for that data wherever it may reside.

The ability to use common MIBs as a translation and description mechanism reduces the amount of traffic required to effect a transfer of information. It is possible to simply ask the remote agent to return the value in OID 1.3.6.1.2.1.1.6.0, for instance. The remote agent then simply returns the OID and the value. There is no need for any further metadata to be passed since

8 Relational Database Management System.

9 It is possible to introduce a query layer on top of SNMP/MIB that is capable of performing such things, but SNMP and the MIB construct only know how to return the variable contained by a given OID.

both sides share a common understanding of the data context. If the value of the MIB variable in question is a Boolean, that means that the entire content of the SNMP packet is one bit.

Of course this does not take into account the overhead for the UDP/IP headers or the SNMP header. 34 bytes are lost to Ethernet and IP headers and 36 more to the SNMP header. Each number in the OID requires at least 1 more byte as well, meaning that to get a single bit of information (in the Shannon sense) we generally need to transmit, on the order of 80-100 bytes of data. Even with such overhead, this makes for very efficient messaging compared to the minimum 1K message size to accomplish the same retrieval via XML/Web Services (Pras, et al., 2004).

From a manageability standpoint, SNMP MIBs are favorable to web services as well. Because SNMP utilizes UDP there is no need for the 3-way handshake required by TCP (including HTTP) connections, cutting down further on overhead. Also, since SNMP utilizes port 161 for get/set messages and 162 for traps, it is much easier to identify this traffic on the network, including setting quality of service thresholds. Web services, on the other hand, look like any other HTTP (port 80) traffic, and require much more sophisticated methods to identify critical web services from normal, lower priority, web traffic.

1. MIB-2

The so-called[10] MIB-2 is defined in IETF RFC 1213 (McLoghrie and Rose, 1991) and contains the general network management information utilized by and required for SNMP management. This MIB is the one likely to be most familiar to network managers. Because of this we have chosen to use it as an example MIB, from which generalizations can be drawn. Sections of this MIB are reproduced in whole or in part from RFC 1213, but in many cases, comments have been removed to ease readability.

[10] Originally, OID 1.3.6.1.2.1 was assigned to the Internet Management Information Base, MIB-I, as defined in RFC 1066 (1988). However, when some content of the MIB was updated (initially in RFC 1158 and later in 1213), a name change was required to indicate a difference between the versions. Since this was the second version of the Internet MIB, the I in MIB-I (which stood for Internet) was retroactively assigned as the Roman numeral 1, making MIB-II, or MIB-2 a natural name for this second version.

The header of RFC 1213 follows:

```
RFC1213-MIB DEFINITIONS ::= BEGIN
IMPORTS
      mgmt, NetworkAddress, IpAddress, Counter, Gauge,
               TimeTicks
   FROM RFC1155-SMI
  OBJECT-TYPE
    FROM RFC-1212;
mib-2      OBJECT IDENTIFIER ::= { mgmt 1 }
-- textual conventions
DisplayString ::=
     OCTET STRING
PhysAddress ::=
     OCTET STRING
-- groups in MIB-II
system       OBJECT IDENTIFIER ::= { mib-2 1 }
interfaces   OBJECT IDENTIFIER ::= { mib-2 2 }
at           OBJECT IDENTIFIER ::= { mib-2 3 }
ip           OBJECT IDENTIFIER ::= { mib-2 4 }
icmp         OBJECT IDENTIFIER ::= { mib-2 5 }
tcp          OBJECT IDENTIFIER ::= { mib-2 6 }
udp          OBJECT IDENTIFIER ::= { mib-2 7 }
egp          OBJECT IDENTIFIER ::= { mib-2 8 }
-- historical (some say hysterical)
-- cmot      OBJECT IDENTIFIER ::= { mib-2 9 }
transmission OBJECT IDENTIFIER ::= { mib-2 10 }
snmp         OBJECT IDENTIFIER ::= { mib-2 11 }
```

MIB-2 begins with an IMPORTS keyword, referencing information available in two different ASN.1 documents. This is obvious in the first definition, where mib-2 is given the OID of { mgmt 1 }. The OID for mgmt (and ultimately this entire MIB tree) is not defined anywhere in this document, but is, instead imported from RFC 1155-SMI (Structure of Management Information). Two new data types are defined, both as strings of octets. What then follows is the list of child OID nodes of mib-2. Each of these is defined by its own OID under mib-2 (tcp, for instance is {mib-2 6}).

We will now follow one of these child nodes, referred to as groups, for further expansion. Since it shows the flexibility of ASN.1 and the MIB architecture, and since it is fairly simple in structure, a portion of the System group { mib-2 1 } is duplicated below:

```
 sysUpTime OBJECT-TYPE
    SYNTAX  TimeTicks
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
       "The time (in hundredths of a second) since the
       network management portion of the system was last
       re-initialized."
     ::= { system 3 }

sysContact OBJECT-TYPE
    SYNTAX  DisplayString (SIZE (0..255))
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
       "The textual identification of the contact person
       for this managed node, together with information
       on how to contact this person."
    ::= { system 4 }
```

These two examples show the basic format of a MIB variable in SNMP. Each variable must be defined with a syntax, an access type, a status, a description and an OID. For sysUpTime, the syntax is given as TimeTicks. This is defined (elsewhere) as a scalar integer; as noted in the description, this is basically a counter of the number of hundredths of a second that have elapsed since boot. The access type for this variable is "read-only," meaning that the variable cannot be set via SNMP `set` commands regardless of the community string in use.

The status field is primarily for implementers. If a vendor intends to implement MIB-2 on one of their devices, they must implement this variable, since it is mandatory. The description field is a free text block where specifics about the MIB variable are shared. This field gives the SNMP MIB architecture a self-describing capability that is extremely flexible. Any arbitrary data mapping can be used in a variable to meet certain needs (brevity, obscurity) as long as the MIB description field is available to the receiver so that they may parse it appropriately. This variable is a good example. A reasonable response for sysUpTime is a large number like 1063188534 (the current uptime on my server at home). Without the description field, that would be a completely nonsensical number.

Finally, this object is given an OID of { system 3 }, which is equivalent to { mgmt 1 1 3 } or any of several ways it could be written. The next variable in the group { system 4 }, also mandatory, is sysContact. As we can see from the description, this is a field that can be used to record contact information for the person or group responsible for a device. It would be possible, then to alert the responsible human if an outage is sensed. This variable, too, is defined as being read-write. This allows us to remotely change the value of this MIB variable, presuming that we have the appropriate community string information.

One concept that is not conveyed from these two examples is that of a table of variables. The ASN.1 language includes a SEQUENCE OF keyword similar to the SET keyword used above. This keyword defines the MIB variable in question as a being multiple copies of something. By using that, we can create a kind of one-column table. A SEQUENCE OF sysUpTimes (although not defined) would create another MIB variable that held a number of sysUpTime values—perhaps we found some reason to maintain the last 5 reboot intervals, for some reason, and found a way to save them as this MIB variable.

By creating a sequence of MIB variables that are themselves SETs or SEQUENCES, we can define 2-dimensional tables. (In fact, this could be extended to n-dimensionality, but as we will see in the example, it becomes remarkably confusing even at 2D). An example is given below, utilizing the IP address table definition from MIB-2. One should note the use of an INDEX field in the ipAddrEntry definition. This index is used precisely as a way to refer to rows in the table.

```
          -- the IP address table

     --  The  IP  address  table  contains  this  entity's  IP
addressing
     -- information.

       ipAddrTable OBJECT-TYPE
          SYNTAX  SEQUENCE OF IpAddrEntry
          ACCESS  not-accessible
          STATUS  mandatory
          DESCRIPTION
           "The table of addressing information relevant to
           this entity's IP addresses."
          ::= { ip 20 }

       ipAddrEntry OBJECT-TYPE
          SYNTAX  IpAddrEntry
          ACCESS  not-accessible
          STATUS  mandatory
          DESCRIPTION
           "The addressing information for one of this
           entity's IP addresses."
          INDEX   { ipAdEntAddr }
          ::= { ipAddrTable 1 }

      IpAddrEntry ::=
          SEQUENCE {
              ipAdEntAddr
                  IpAddress,
              ipAdEntIfIndex
                  INTEGER,
              ipAdEntNetMask
                  IpAddress,
              ipAdEntBcastAddr
                  INTEGER,
              ipAdEntReasmMaxSize
                  INTEGER (0..65535)
            }
```

2. Other Uses for MIBs

While it is certainly possible to exchange SNMP data without sharing a common MIB, the result is unformatted and undescribed data being transferred. Conversely, when both sides share a common MIB, it is possible to transfer any arbitrary data desired. Even the most general case, unformatted plain text messages, can be exchanged via SNMP by assigning those text messages to a MIB variable on one machine and retrieving that OID from the other. This information need not be network management type data, nor need it refer only to technical systems or devices.

While this seems a bit clumsy as described, and would be in practice, imagine instead that there is a defined MIB variable called, "statusMessage." Instead of Boolean or integer status definition as above, we define statusMessage as a character string. Now, a user on the monitored device can input in free text a status message referring to the machine, himself, or even the organization he represents (working, but just barely; out to lunch; closed for the three-day weekend). A monitoring station would receive that status message the next time it retrieved the SNMP information. More powerfully, the agent could be set to send an SNMP trap whenever that status changed, alerting the management station to the change. This concept and variations upon it as proposed in this thesis can empower Hypernodes to exchange vast amounts of data conveying information at all levels of the O-I-T stack.

THIS PAGE INTENTIONALLY LEFT BLANK

VII. HYPERNODES AND THEIR MIBS

A. HYPERNODES

We will now look to extend the concepts of managed devices and management stations with a third class of SNMP-connected nodes: Hypernodes. The original concept of Hyper-nodes derives from Bordetsky and Hayes-Roth (2006), wherein they suggest that the 7-layer OSI model requires an additional, 8th, layer, comprised of network-management-aware functions. Those network nodes which are 8th-layer aware, then, are Hyper-nodes. The 8th layer of Bordetsky and Hayes-Roth, it must be clear, is not the same as the 8th layer discussed in Bauer and Patrick (2004) and adopted into the O-I-T model. Particulars of terminology aside, their fundamental argument is sustained and is extended by this thesis.

Hypernodes[11], in this case, consist of three different classes of network devices: those which are network service aware (in the case of Bordetsky and Hayes-Roth, the specific service is network management), those that are subnetwork aware and those that are decision support aware. While these three classes seem widely divergent, in truth, they all share significant commonalities. All will be implemented in an SNMP architecture, and all utilize conceptual extensions to SNMP to facilitate the additional functionality. Additionally, none of these extensions needs to affect the underlying network, SNMP standards or any network nodes which are not, themselves, Hypernodes.

We propose the use of MIB space within the US DoD OID hierarchy. Unlike the Internet portion of the DoD's OID space (familiar to network managers as 1.3.6.1), general DoD applications are allotted space within the 2.16.840.1.101.2 (joint-iso-ccitt (2) country (16) us (840) organization (1) us-government (101) dod (2)) tree. As the first and second children of this tree are already in use for infosec (1) and X.500 (2), we propose to use 2.16.840.1.101.2.3 as the Hypernode space. Unfortunately, we have been

[11] I choose to use Hypernodes instead of Hyper-nodes in this thesis to differentiate them from the original Bordetsky and Hayes-Roth construct.

unable to receive a grant of space in this portion of the tree. An OID has been applied for in the 1.3.6.1.4.1 space as well.

1. Network Service Aware Hypernodes

One problem that often vexes network users is the need to find and access network services. We know, for instance, that on the Global Information Grid (GIG) there are a large number of systems providing weather databases. We also might suspect that there are a number of locations where we might go to find the status of the network or where imagery of a certain geographic region might be found. These network services are the heart of the GIG concept, but quite often remain unknown and unaccessed simply because there is no uniform manner for finding and gaining access to these services.

In a service-oriented architecture built on Web Services, we would implement search via UDDI (Universal Description, Discovery and Integration) and expose the services themselves as web services. This means, though, implementing an entire web services stack on every service-providing asset in the network. While this is not a particularly difficult task for many of the "back-office" sorts of nodes found in datacenters ashore and removed from the front lines, it is fairly onerous if one considers that every node is potentially a service provider. These nodes may have very few slack resources to devote to secondary tasks, and the addition of such infrastructure may overwhelm the computing capabilities available on a UAV or to a dismounted infantryman.

Utilizing SNMP, however, will often allow such a service architecture to be implemented without having to resort to any extra software on the end nodes. As mentioned before, nearly all network-attached devices have or have available SNMP agents. The addition of Hypernode functionality to them requires only the development of a Management Information Base (MIB). These MIBs might be specific to the functionality provided or could be fairly generic in the cases where custom development is uncalled-for. Since the MIB, as mentioned previously, is merely a database schema, the addition of such functionality would represent relatively little in the way of static resources (hard disk, for example) used and would only require dynamic resources (RAM, CPU) when being accessed.

A conventional SNMP-managed node is not aware of the fact that it network management is, in fact, ongoing. Nor is its SNMP process aware of any other services that are being performed by the node. There is, though, some ability to report upon those services. Some application MIBs have been developed to allow for remote management and monitoring of those applications. The Oracle-database-MIB is a good example (1.3.6.1.4.1.111.4). With this MIB, it is possible to retrieve variables related to the operation of the Oracle database such as the number of user calls or the number of user commits (1.3.6.1.4.1.111.4.1.1.1.22 and .23 respectively).

This does not, however, tell us what is in the database or how to access it. What is needed is a new extension to the commonly used SNMP MIBs that makes such information available. A network service aware Hypernode meets this requirement by containing within its MIBs an explicit description of the services provided by the node. The same Oracle database, then, in addition to any generic RDBMS or Oracle-specific SNMP data would expose, at least, the nature of such databases and information on how to gain access.

To continue this example, a node which provides a weather database for a specific geographic region would need to advertise this service somehow. Its MIB should indicate 1) that it is a service provider; 2) that it provides weather data; 3) for which geographic region it has data; 4) how to go about retrieving the data. It might, additionally, include other meta-information about the service it provides, such as the timeliness of updates or, in this case, whether it provides aeronautical weather information or nautical or general.

Bordetsky and Hayes-Roth (2006) suggest this treatment for network management as well. If a node is capable of providing network management capabilities, these should be represented in that device's MIB. This might include simple capabilities, like a wireless access point advertising that it is, in fact, a wireless access point—that is, a network node which provides the service of extending the network wirelessly. Or more complex capabilities, such as a node which streams video and is capable of modifying the output data rate (thereby managing the amount of data traversing the network). When all these nodes advertise these services via SNMP and, to the extent possible, allow them to be modified via SNMP, they are network service aware Hypernodes.

a. Extending the Network

To illustrate how such a concept might be realized, consider a simple subnetwork made up of only three nodes. One node is our dismounted infantryman, perhaps a special operator, who is carrying a network-attached sensor capable of multi-spectral imaging. The second node is back at base camp, a standard PC, connected via a robust network link back to the rest of the GIG and being monitored by a local decision-maker in need of the imaging data our SOF operator is acquiring. The third network node is a UAV. As long as the SOF operator is within a reasonable distance of the base camp, he can continue to send imagery and the decision-maker has access to his services. See Figure 11.

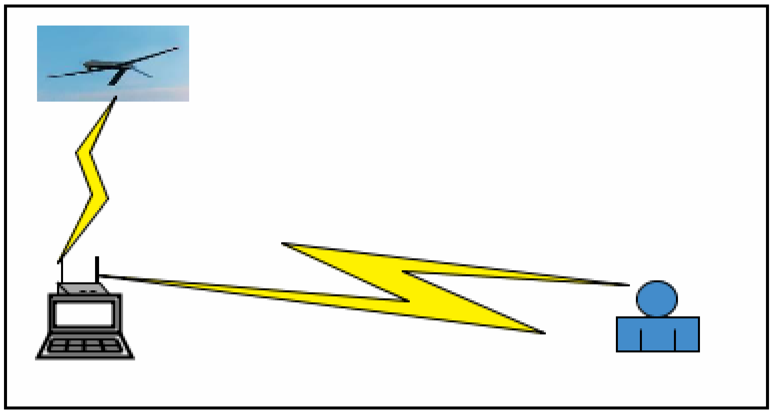


Figure 11. Line of Sight

However, as he moves farther away, the radio can no longer establish communications (or more likely is no longer capable of providing sufficient available capacity to meet the requirements of the imaging). In the case of Figure 12, our network service aware Hypernodes can make this information known to both ends. The SOF operator might see this diminished capacity and attempt to relocate or reduce the demand created by his activity (such as reducing the refresh rate or resolution of his imagery). The decision-maker might attempt to increase his transmit strength (via SNMP commands to the transmitter!) If he can accept degraded video quality, he might make the same attempts to reduce demand as the SOF operator.

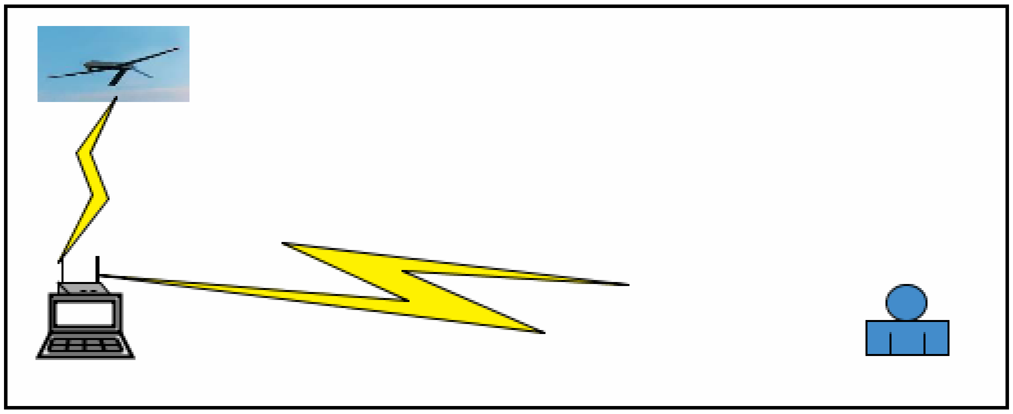


Figure 12. Loss of Line of Sight

This will not always solve all the problems, however. What may be needed in this situation is additional capacity as opposed to reduced demand or a simple repositioning.

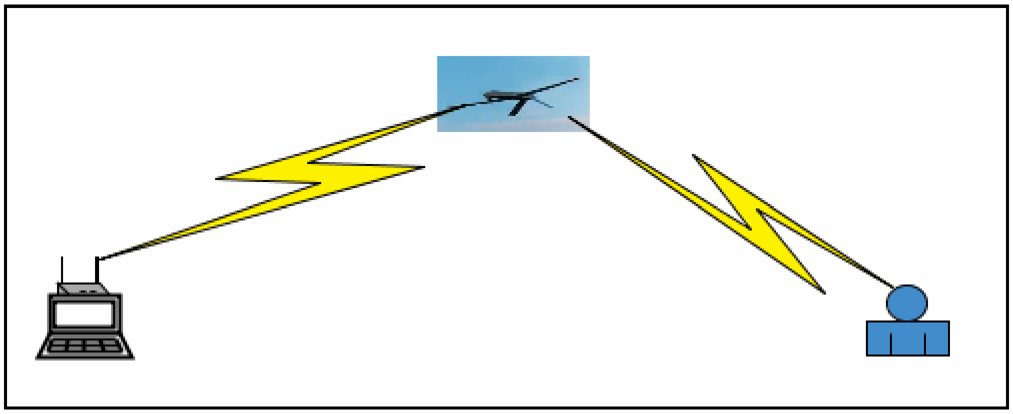


Figure 13. Link Restored

Fortunately, the UAV in the area is also a network service aware Hypernode. This UAV can be used to relay radio communications to the SOF operator. When the decision-maker at base camp searches for a node with the ability to extend his network, he finds the UAV and sends it out toward the SOF operator as shown in Figure 13. If the UAV is particularly sophisticated, it might be able to process a request such as "maintain 300Kbps on this link," and do what maneuvers are necessary to maintain that as conditions and the location of the SOF operator change. Even without that capability, though, the decision-maker can continue to adjust the location of the UAV to maintain such link capabilities in a human-in-the-loop manner. As we will see below, we may also wish to model the services provided by that human.

A more sophisticated multiple-criteria problem exists, however if that UAV is, unlike this simple case, attempting to maintain several network connections throughout the battlespace. If the UAV is, in fact, a sophisticated Hypernode, as described above, it might attempt to solve the problem itself, alerting human users only when it can no longer communicate. If it is unsophisticated, the humans would need to arbitrate for themselves or at higher headquarters. Here, too, Hypernodes can play a role. Since the base camp is ultimately performing a service—whatever his mission is—this, too, can be entered as a service in the SNMP MIB on his computer. Now, instead of querying all the entities vying for access to the UAV, higher headquarters need only query their Hypernodes.

b. Hypernodes Representing Humans or Groups

This is an additional potential for SNMP Hypernodes and should not be overlooked. In the usual fashion, SNMP agents only report on themselves. They return requests for data about the hardware and the software processes running on them. There is no reason, however, that this must be so. If an entity—a command, a unit, a fire team, an external expert—is capable of providing services to the network, they can be represented in a MIB.

In this case, the SNMP agent must still run on some network-attached computer (at least until such time as we are, ourselves, network attached computers). Any computer will do, but only one should be identified as the Hypernode representing a Human or Group (of course, there is no reason that a single computer could not be the Hypernode for more than one). An obvious choice might be for the command's web server to function as the

Hypernode for that command. The SNMP MIB for this command would then enumerate the capabilities of the command and point to the Hypernodes for any subordinate commands.

When describing Hypernodes which represent humans and groups, another interesting possibility is suggested. Humans and groups of humans tend to develop relationships with one another. These relationships, themselves, become services that can be accessed on the network. Hence, service aware Hypernodes should also be relationship aware in the case of humans. By expressing these relationships in a MIB, it becomes possible to look at a network of Hypernodes connected by relationships as having emergent capabilities themselves. Further, the modification or addition of relationships could be used to reconfigure such a network for other or more varied tasks.

In the next few chapters, we will be dealing with one example of a network into which the addition of SNMP Hypernodes is suggested. In that case, there are a number of external experts connected to the network. These non-military entities may be unknown to the participants in the network, but may have crucial services to provide. Establishing a network service aware Hypernode for these experts allows other network members to either query for required services or investigate the services provided by these nodes.

In order to facilitate this kind of search, and the others mentioned in this thesis, some subset of a universally agreed-upon taxonomy will be required to name the provided services in meaningful ways and to allow for informed searches. The development of such a taxonomy is beyond the scope of this thesis, but an obvious start is provided by the DoD XML registry: http://metadata.dod.mil. The same taxonomy used by the DoD to describe XML tags may be useful for our purposes.

2. Subnetwork Aware Hypernodes

The SNMP specification (Case, et al., 1990) specifically allows for the creation of proxy agents. According to the specification, these proxies might be used to either translate SNMP requests and responses to and from a second protocol or to forward messages to nodes that are not addressable using the transport protocol used by the Network Management Software (NMS). While powerful, in the first case allowing us to manage non-SNMP enabled devices with SNMP and in the second case allowing us to traverse transport protocols (perhaps we need to manage devices on an AppleTalk segment from a TCP/IP network), in both cases the proxy agent is expected to merely translate and forward the request to the destination agent.

We will discuss Hypernodes repeatedly as those SNMP nodes which serve specific functions on a network. There need not be anything more unique to Hypernodes than that they implement portions of the Hypernode MIB—there is no other hardware or software that is required for a node to be a Hypernode. For the specific case of subnetwork aware Hypernodes, those devices for which they are responsible will be referred to as child nodes. While there is no specific technical reason to enforce such a rule, it is suggested that a network node be a child of only one Hypernode at a time. It is possible, of course, to simply remove duplicate information after aggregation at a higher level, but this requires overuse of transmission resources and adds processing complexity.

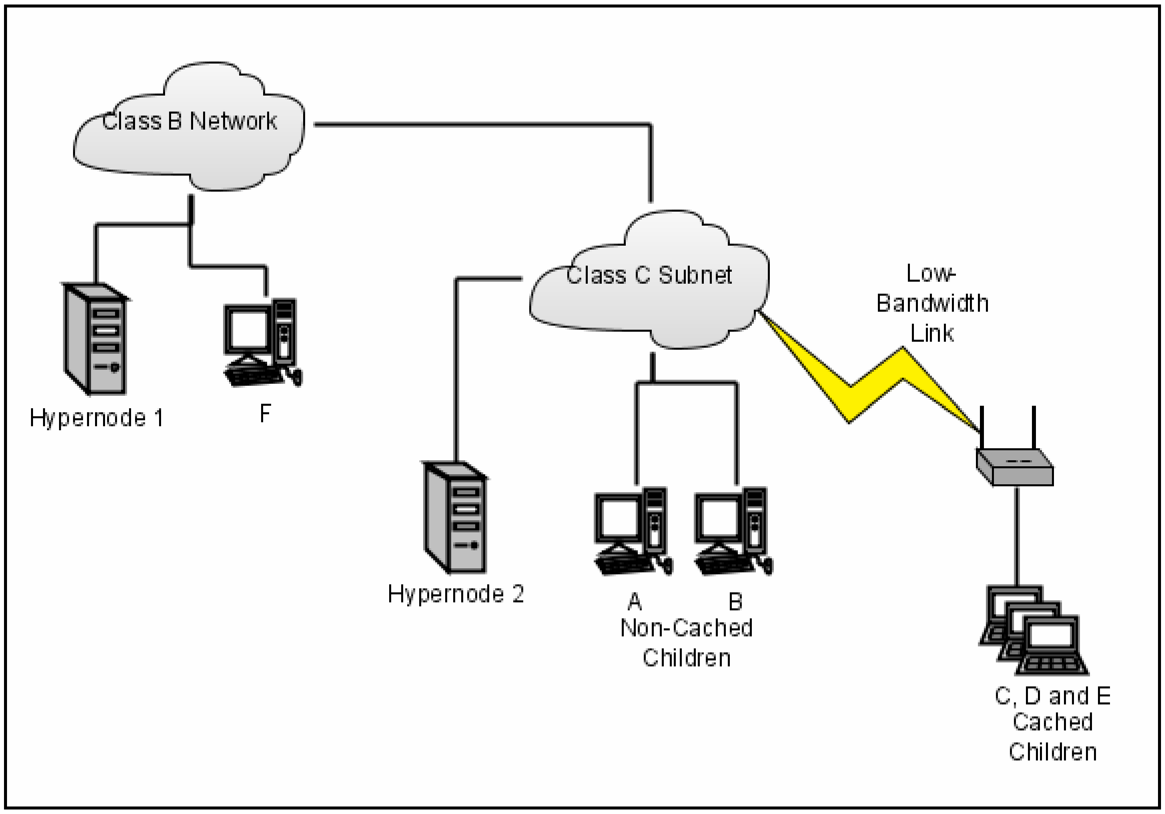


Figure 14. A Network with Hypernodes.

This is not to suggest introduction of a single point of failure in a subnetwork. Contrarily, Hypernodes may belong to a cluster, allowing them to share all their collected information and respond on behalf of any node in the cluster in case of a failure or network segmentation. Cluster membership information will be shared with both child nodes and higher-level Hypernodes to allow for redundancy, fault tolerance and automated failover. Figure 14 describes one possible small network with subnetwork aware Hypernodes.

a. Overcoming Bandwidth Constraints

If we consider a network where some of the nodes are on bandwidth constrained links, such as low data rate radios, simply forwarding SNMP requests can quickly saturate the entire capability of a link with just management traffic. What is suggested as a solution in this case is a caching proxy. A subnetwork aware Hypernode is precisely this sort of a proxy. The bandwidth-limited nodes could be configured to only communicate via SNMP trap messages to the Hypernode at some interval appropriate for their communications link. When the Hypernode receives an SNMP request for the subnetwork for which it is responsible, it answers the request itself, replying with data from the cached trap messages.

The details of such an implementation are, in some cases left for future research, but a few guidelines are proposed. Hypernodes should contain a table of SNMP MIB values/OID pairs under the Hypernode OID tree. Any MIB data (both network management and services types) that a child node sends to a subnetwork aware Hypernode will be saved into that portion of the Hypernode's MIB. A simple Boolean value will also be included to signify whether the Hypernode should answer on behalf of the child device or not. When a Hypernode is queried, then, it will respond with MIB data on behalf of those children it is instructed to and respond with addresses of those children which desire to be polled directly.

The obvious advantage over other methods is the homogeneity of protocols allowed by SNMP Hypernodes. Currently, such a proxy caching system as described above requires a separate protocol to be utilized for communications between the top-level management application and the proxy nodes. This severely constrains the choices available for network management software. Worse, the protocols implemented by most large management systems (those which have the ability to aggregate and cache) such as Tivoli and OpenView, are not available outside those tools. If we wish to use any of the information available in these subnetwork aware proxies, we must also use their software. When we look to expand beyond network management functions and share that information to arbitrary users in the network, the requirement to use a proprietary protocol is truly onerous.

b. The Large Network Problem

Another concern involves dynamic subnetworks. SNMP, in general, relies on some external method to discover network nodes. One common procedure is as follows:

1. A network operator is alerted that a new device may be on the network, or that a new network segment has been connected.

2. If the address (range) of the device(s) is known, ICMP Echo Requests (pings) are sent to the address range in question. If the address (range) is unknown, ICMP Echo Requests are sent to the entire network.

3. Any device that responds and is not in the current database of managed nodes is then sent a series of SNMP `get` commands to return the contents of the system MIB. In order to do this, some number of community strings must be tried, either based on default configurations, established procedures or specific knowledge.

4. Devices that respond to SNMP are further queried for details the network manager is interested in (the interfaces MIB, for instance) and are added to the management system.

5. Devices that fail to respond to SNMP are (perhaps) added to the management system as unmanaged hosts. Status will be monitored by using ICMP and looking for responses to indicate an operational status.

This procedure is fairly straightforward and is used in one incarnation or another in many NOCs every day. Unfortunately it is not very useful in a highly dynamic environment. A few numbers will help to illustrate this situation. If we consider a very small (class C) network of 254 possible nodes, and allow 0.5 seconds per node to respond to the ping and SNMP requests outlined above, it will take a little over 2 minutes to discover and register the entire network. This assumes, of course, that every device responds. If we have to wait for requests to nonexistent or non-SNMP enabled devices to time out (usually about 2 seconds), this can easily stretch toward 8-10 minutes.

Most network discovery tools are capable of issuing multiple requests in parallel, consequently, it is not necessary to wait the full two seconds for one node to timeout before proceeding to the next one, nor must we even wait for the 0.5 seconds that responding nodes require. For the moderately populated TNT network that will be discussed later (about 40% of the addresses are actually active nodes), this network discovery process occurs in about 35 seconds for a single class C network. If we know, then, that a node has been connected in a specific class C, or we know that an entire class C network has been attached, it should only take about a half a minute to discover it and potentially add it to our management system. This is a perfectly manageable timescale for discovery in relatively static networks of this size.

If, however, our network is a Class B (65,534 addresses) and we do not know what address a new device has taken, the prognosis looks more bleak. At the same rates as above, it will now take nearly 2 hours to scan the whole network. Given the fact that the DoD is issued an entire Class A network (16,777,214 nodes) and that many other Class A and B networks are further connected to it via such tools as Network Address Translation (NAT) finding a newly connected node could take an impossibly long time. When networks are changing on a rapid basis, such as with networked aircraft transiting an area of operations, or mobile ground forces moving from one network coverage area to another, such a long lead time to find new nodes is untenable.

Transition to IPv6 (Internet Protocol version 6) only exacerbates this problem, since the smallest network in IPv6 consists of approximately 2^{64} (1.8×10^{19}) addresses, and networks will, by default, be assigned an address space with room for 2^{80} (1.2×10^{24}) addresses. Even if we can find and register 1000 nodes per second, the smaller of those address ranges would require more than *1/2 a billion years* to fully discover[12]. Clearly our existing procedures for discovering and adding nodes to our management system do not scale well or deal with rapidly changing networks.

Subnetwork aware Hypernodes also address this problem. In addition to caching and allowing proxy retrieval of connected network device information, subnetwork aware Hypernodes contain information about which devices are in their network and for which address range they are responsible. Every device in the subnetwork, further, contains the address of its Hypernode. As a consequence, discovering *any* node in a subnetwork gives enough

12 In case you were wondering, the larger (default) network size could be completely discovered in just over *38 billion years* if we were able to find and register, instead of 1,000, 1 million nodes per second.

information to find all the remaining nodes. If we make the simple assumption that we must know (or can easily determine) one address, that of the router interface on that subnetwork, the problem of enumerating this large network vanishes.

It is important to note that the primary problem we are attempting to overcome here is the need to do a blind search in a sparsely populated network. If we have some way of informing our search, or if we know that a subnetwork is, in fact, fully or nearly fully populated, the results from a search are much improved. Unfortunately, in such a case, we are still faced with interrogating every node to determine the state of the network. Aside from offering a solution to the sparse network problem, if our subnetwork aware Hypernodes are caching management data, we need only interrogate one, presumably fast and well-connected, node, although we will need to interrogate it intensively.

The following flowchart attempts to describe this process at a high level. Upon discovering a node in a subnetwork, we first request the MIB variable which identifies the node as a Hypernode. If the result is positive, then we know the Hypernode address and simply ask for all its data. If the result of the query is negative (not a Hypernode) we then request the address of the Hypernode to which this node belongs. We then repeat our "is Hypernode?" query on the presumed Hypernode. Assuming a positive response (it should be) we then retrieve the subnetwork information.

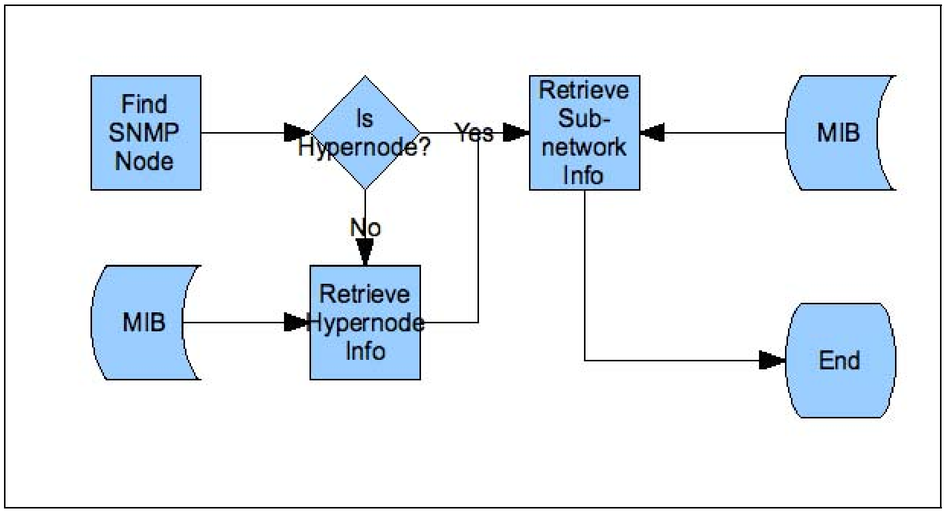


Figure 15. Subnetwork Discovery Flow

New nodes joining a subnetwork (including subnetwork aware Hypernodes whose subnetwork is joining a larger network), then, should first seek out their local subnetwork's Hypernode and register with it. The Hypernode will then forward this information upward as required. Other devices in the network, looking to discover devices or services, need only find Hypernodes (at least initially) within each subnetwork. Given the above process, it can be seen that we have reduced the problem of completely discovering a network address range to a problem of discovering any one SNMP device within it.

With an appropriate Hierarchy and forwarding rules, finding all nodes in the entire network is possible by finding any registered node in the entire network. Appropriate forwarding rules ensure that Hypernodes at the root of the network do not have to cache unnecessary large volumes of data, but that bandwidth-constrained nodes need not be concerned with constant polling.

3. Decision Support Aware Hypernodes

The final class of Hypernode is strongly related to the network service aware Hypernode. These Hypernodes represent a special class of network service: decision support. As we will see in the following chapters, one of the primary uses of collaborative technologies we have discovered is to share the results of and request the status of decisions. Unfortunately, making and answering such requests often represents a significant time commitment on the part of all parties involved. By utilizing SNMP MIBs to store these decision states, it is possible to offload much of this work onto computers.

Decision-makers could choose to use SNMP traps to immediately distribute information about a time-sensitive or anticipated decision, while other users might periodically poll, using get commands to retrieve desired information. The implications for the sharing of decisions up and down the chain of command are fairly obvious and could even take advantage of subnetwork aware Hypernodes in order to optimize the use of scarce network resources where necessary in transferring these decisions.

a. Hypernodes in Edge Organizations

Potentially even more useful is the ability to look across the organization at decisions that peer or unrelated units have made. If we assume that, moving forward, our military activities are going to require more work with allied, coalition and non-governmental partners, as encompassed by the renewed focus of Military Operations Other than War, (JCS, 1995) the maintenance of information about decision states ceases to be synonymous with military command and control and, instead, becomes an enabler of more Edge-like organizations (Alberts and Hayes, 2003) as groups and individuals anywhere within the organization can immediately access the decision states of all the other involved groups.

Edge organizations, according to Alberts and Hayes allow for leaders to emerge due to their capabilities or expertise in a given situation. Information flow is generally unrestricted, allowing "appropriate interactions between and among any and all members." (2003) This is in sharp contrast to the traditional military hierarchy where information flow is constrained by tradition and by fiat and where leadership is primarily a function of position.[13] Decision

13 That is not to say that that position was not gained by expertise, or the "sustained superior performance" expected of military leaders. What is suggested is that the specific task, situation or mission may call for leadership skills and expertise not held by such an appointed leader.

rights, too, are to be distributed to the lowest level possible, allowing individual nodes to make as many decisions locally as possible.

While not a perfect description of the situations created when the military must work with outside agencies, the Edge is a much better description than a military hierarchy, and offers a suggestion on where we should, perhaps, be headed. Hypernodes offer one path for both push and smart-pull information architectures that are required by VIRT and Edge organizational forms and empower network members with the information they need to make decisions at their level while understanding not just the overall "commander's intent," but also the changing decision states in the network.

As an example, imagine a multi-agency response to a natural disaster. Organizations as varied as FEMA, the Red Cross, the U.S. military, religious aid organizations and local fire and police will, if history serves as any guide, all be involved in the resolution of the disaster. These various organizations will all be making decisions based on the desires of their constituencies or their superiors, and while they are all working toward the same broad goal, the way they conceive of it may be quite different; what the Red Cross expects to do to support a natural disaster may or may not be in line with what the other organizations believe is the appropriate course of action.

We cannot expect these agencies to defer to one another, nor can we expect them to know with whom they should discuss their plans a priori. By utilizing decision support aware Hypernodes, however, these agencies can simply "post" their decisions and allow others to poll them. When the Red Cross decides to set up a relief supply distribution site, they can enter such a decision, as well as its location and any details they feel germane (types of supplies, what hours it will operate) into their organizational Hypernode for others to find. If, and likely when, they are queried for further information, they can simply update the existing information or choose to include more on their Hypernode.

If the agency believes it is important enough that others need to know the status of a decision, they could, instead choose to broadcast this to all involved parties via an SNMP `trap`. Astute readers will, of course, insist that this could all be handled via other means such as email or web pages. While this is true, the use of Hypernodes is superior in several respects.

No one needs to install/configure/manage web or email servers. This is not a particularly convincing argument in the case of day-to-day operations, but in a case, such as the one above of disaster relief, where the network (both IT and organizational) is being created in an ad hoc manner, simply deciding whose web and email servers to use is non-trivial. Also, this represents excess resource allocations than SNMP, where the agent software is already available to nearly all network devices.

Web and email messages are not particularly machine-friendly. By using Hypernodes to share decision state, SNMP clients can be instructed to take action based on specific updates. Creating the same rules for, especially, web pages is a significantly greater challenge, as the lack of pro forma information on most web sites resolves to a natural language processing problem. Utilization of, for instance, Real Simple Syndication (RSS) would achieve a closer fit to the capabilities of the SNMP model, but at an even higher resource cost.

In the following chapters, we will be dealing with precisely these types of decisions being made by an ad hoc network of actors responding to a Maritime Interdiction Operation. We will see how they interacted with each other and in relation to the various decision states within the network and will make recommendations about the construction of Hypernodes to achieve more efficient information exchange in the future.

THIS PAGE INTENTIONALLY LEFT BLANK

VIII. THE TNT-MIO EXPERIMENTS

A. TACTICAL NETWORK TOPOLOGY (TNT)

Located at the Naval Postgraduate School in Monterey, California, the Center for Network Innovation and Experimentation (CENETIX) has, since late 2004, been involved in a series of experiments collectively known as "TNT." This campaign of experimentation, carried out under the NPS-SOCOM Field Experimentation program involves quarterly field experiments in which a large number of NPS researchers and students investigate various topics related to networking.

TNT is a follow-on to the STAN (Sensor and Targeting Area Network) series of experimentation and is focused on both technologies associated with networking and the human aspects of networked forms of organization. Technologies investigated have included network-controlled UAVs, various forms of wireless networking, a networked Light Reconnaissance Vehicle (LRV), Deployable Network Operations Center (DNOC) architectures and many more. In all of these experiments, the focus has been on both adopting commercially available technologies to military requirements and on investigating the human elements associated with the addition of such technologies to the battlespace.

1. Mission and Objective

The mission and objective of CENTIX, as stated on the center's website (http://cenetix.nps.edu) are:

> Mission — The mission of CENETIX is to provide students and faculty with opportunities for interdisciplinary study in tactical self-organizing networks, with emphasis on wireless networks, sensors, unmanned vehicles, intelligent agents, and situational awareness platforms.
>
> Objective — CENETIX provides flexible deployable network integration and operating infrastructure for interdisciplinary studies of multiplatform tactical networks, Global Information Grid connectivity, collaborative technologies, situational awareness systems, multi-agent architectures, and management of sensor-unmanned vehicle-decision maker self-organizing environments.

As indicated by the name — Tactical Network Topology — the military requirements under scrutiny are generally of the last mile or tactical nature. As an example, the LRV series of investigations has centered on the construction of a small mobile platform for distributing wireless connectivity from long-haul networks down to the dismounted infantryman. The UAV experiments, meanwhile, have focused on allowing forward operating bases the ability to remotely control simple UAVs over a network to extend their observation range. Various other technologies have also been investigated in order to support the mission of the U.S. Special Operations Command (USSOCOM), who is a major sponsor.

2. Experimental Network

Each quarter, approximately two-thirds of the experimental time is spent in the field at Camp Roberts, California evaluating these technologies and investigating the human impacts. These field experiments are then connected back to the Network Operations Center (NOC) at NPS via an 802.16 network link. All the network infrastructure involved is operated and maintained by students and is, itself, often the subject of some experimental activity. High-level network views of a few of these sites are given in Figures 16 and 17.

Figure 16 shows the fixed wireless 802.16 backbone connecting NPS with Camp Roberts. This image is taken directly from one of our network management applications and shows the status of all the nodes in this segment of the network. Figure 17, while several experiments out of date at this point, is a wonderful illustration of the complexity of the TNT network setup. The LRV is indicated as a mobile OFDM[14] node. Also visible are a number of remote field locations that have been used in past experiments.

[14] Orthogonal Frequency Division Multiplexing — this is the technology that underlies 802.16.

In addition to the locations at Camp Roberts and NPS, various remote sites are connected to the TNT infrastructure via an ever-changing set of Virtual Private Network (VPN) links, satellite links, iridium phones and other technologies. As such, a large portion of each experiment is concerned with the collaboration and coordination necessary to integrate the large number of sites and interested parties into the ongoing activities. A few of the VPN sites connected to TNT are shown in Figure 18.

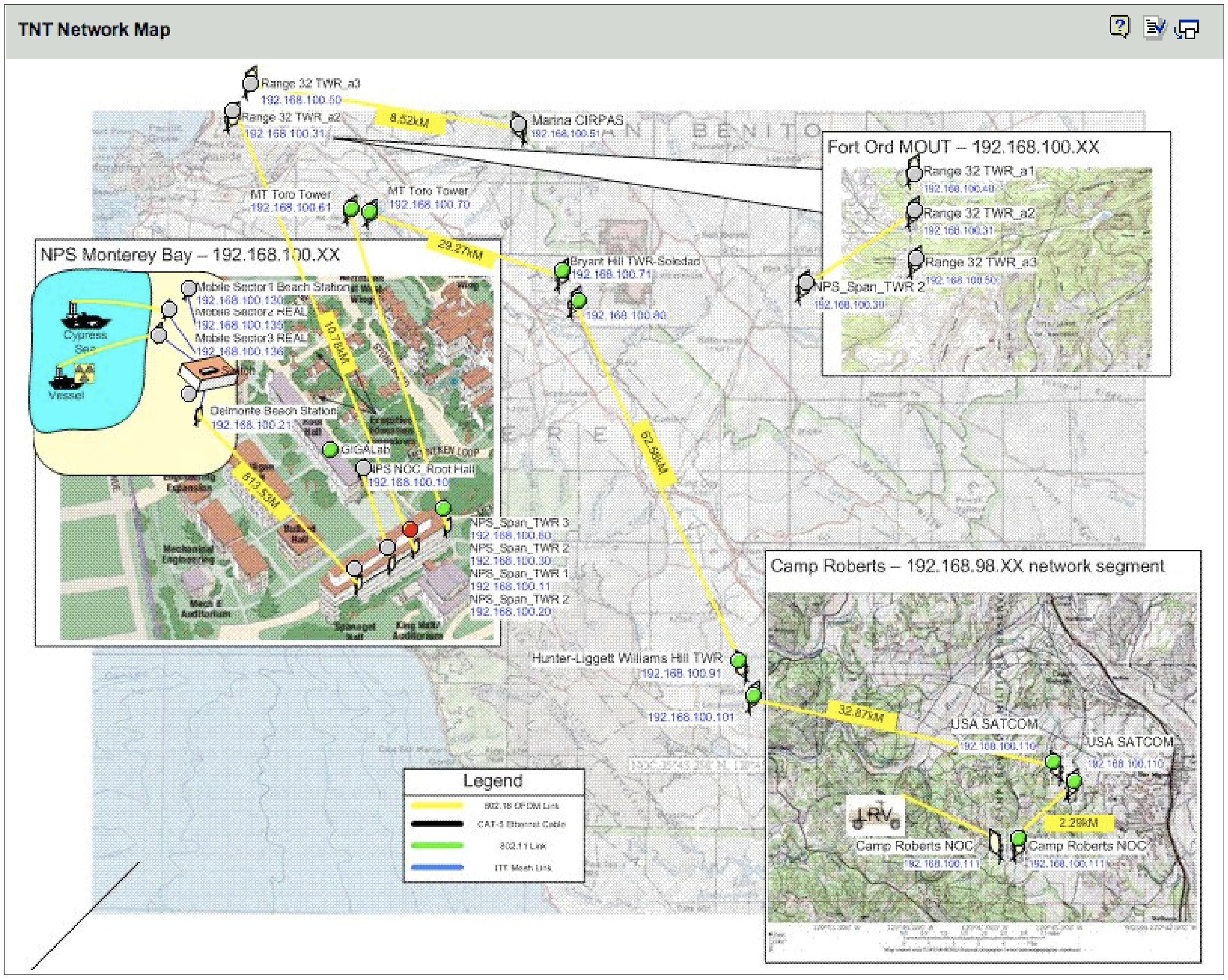


Figure 16. 802.16 Backbone

3. Partners and Sponsors

As they will be involved in the analysis of Chapter IX, it is worth elaborating on several of the nodes labeled in Figure 18. The Biometrics Fusion Center (BFC), located in West Virginia, has been a member of many of our experiments. They are concerned with our research as a way of connecting remote, tactical field users to biometrics databases removed from the battlefield. In this manner, field agents looking for suspected terrorists can take sensors (fingerprint, facial recognition, etc.) directly to the area of interest while drawing on the full (and likely updated) databases provided by the BFC. Conversely, information gained in the field can be immediately made available to analysts back at headquarters or located in other locations around the world.

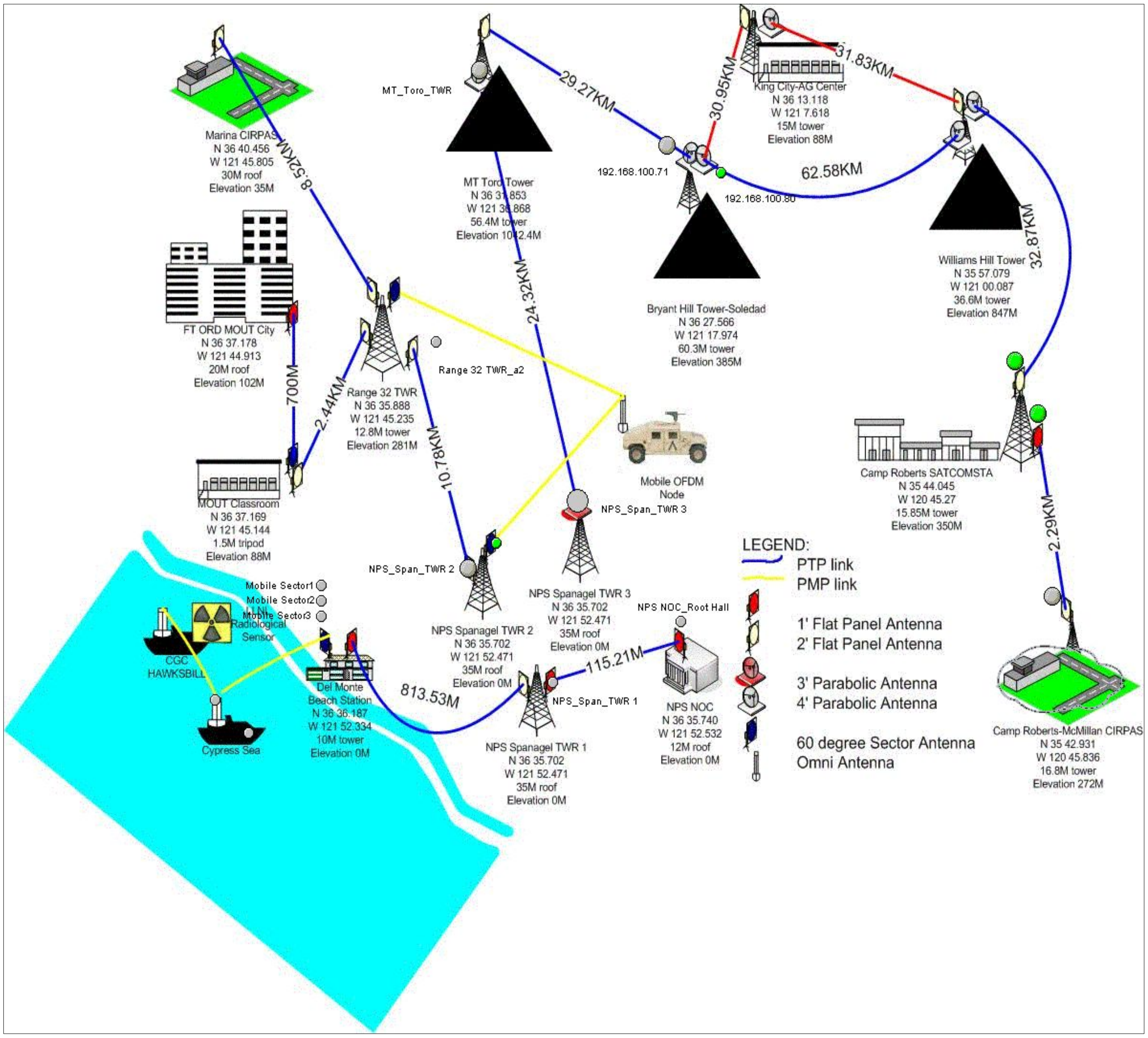


Figure 17. Experimental Network

The Mission Support Center (MSC) is located in San Diego, California. The Navy Special Operations units involved in the TNT experiments access the network from this location. Additionally, the Stiletto ship, which has participated in a number of experiments, is home ported in San Diego and gains access through this VPN

The remaining approximately one-third of the TNT experimentation not occurring at Camp Roberts is involved specifically with Maritime Interdiction Operations (MIO), usually conducted in the San Francisco Bay area. The network infrastructure that supports the MIO portion of TNT will be covered in the next section, but it, too, is connected to the NPS NOC (and to all the other sites) via a VPN link indicated on Figure 18 as NCGS.

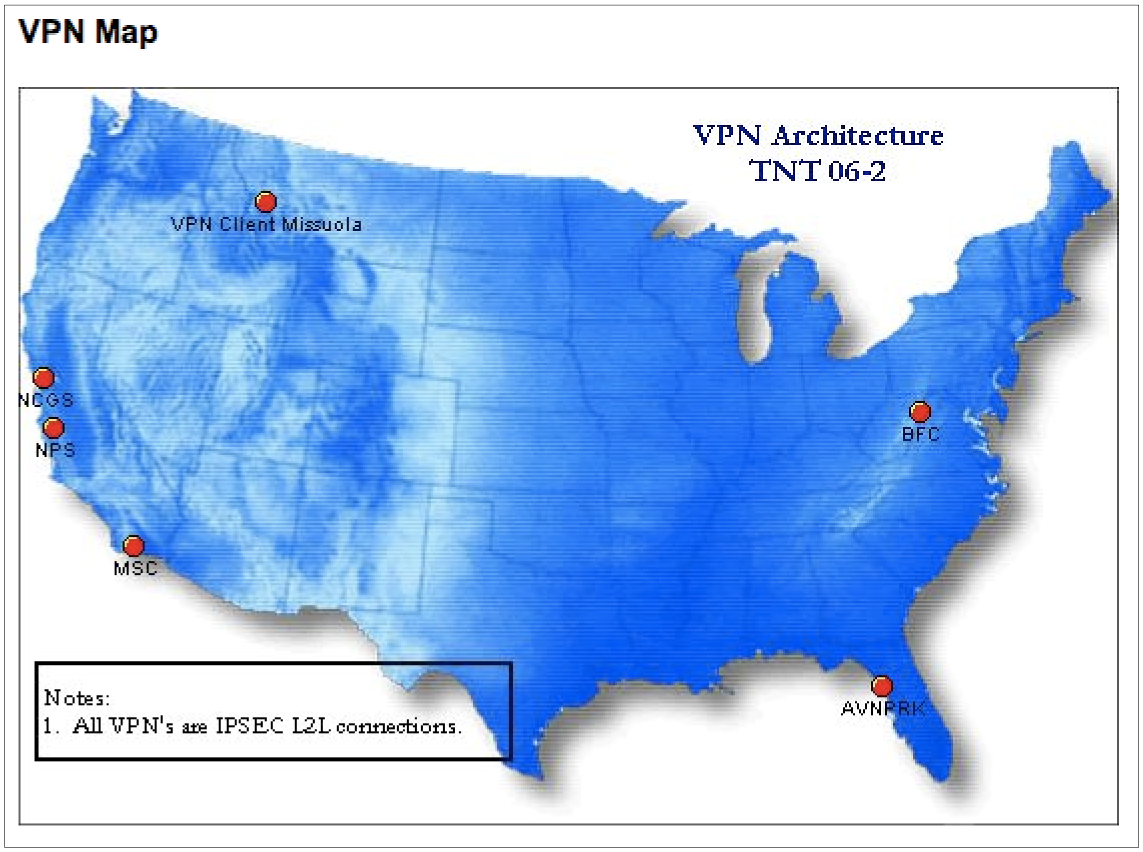


Figure 18. VPN Sites

Not visible on this map are VPN links to Austria, Sweden and Singapore. As many nations in the world are concerned with technologies to increase the effectiveness of the Maritime Interdiction Operations, the TNT test bed continues to gain partners who each bring unique capabilities, technologies and operating procedures. All of these serve to enhance both the quality of experiments and the range of variables under investigation.

B. MARITIME INTERDICTION OPERATIONS

As our focus of investigation, we have chosen the Maritime Interdiction Operations portion of the TNT experiments. This highly dynamic, multi-agency series of experiments provides an excellent case study from which many desirable qualities of Hypernodes can be drawn. Further, appropriate evidence is available to suggest utility for all three of the previously identified classes of Hypernodes.

1. What is MIO?

Joint Pub 3-07 defines MIO[15] as "operations which employ coercive measures to interdict the movement of certain types of designated items into or out of a nation or specified area." (JCS, 1995) Of course, the assumption in the name is that these items are being moved in or out via a maritime avenue (we are not concerned with overland cross-border smuggling, for instance). MIO, then, is primarily comprised of those actions taken to prohibit undesirable shipping to take place. In the current Global War on Terror, this is of obvious concern when we consider the amount of shipping traffic entering and exiting U.S. ports on any given day and the ease with which harmful goods could be brought into the United States.

2. TNT-MIO

The TNT-MIO experiments are focused on several scenarios related to that concern, specifically interdicting the smuggling of chemical/biological and nuclear weapons and nuclear non-proliferation items and identifying known terrorists transiting on container ships. Due to the complexity of this set of tasks and the varied skills required to successfully execute such a mission, our experiments are focused on creating a collaborative network of experts and

[15] Originally, the acronym MIO stood for Maritime Intercept Operations. Recently, the "I" has been repurposed to "Interdiction." This definition was actually written for the former "I," but remains in use.

linking them with the Navy and Coast Guard operators who are engaged in the actual interdiction. These interdictions take the form of a VBSS (Visit, Board, Search and Seizure).

As the TNT-MIO experiments have progressed, we have moved from simulated boardings of large ships pier side to actual boardings of ships in protected waters (San Francisco Bay). Future experiments will increase the fidelity of the operation by moving the VBSS operation out into the open ocean. If we consider the situation where a potential terrorist is attempting to bring a radiological device into an American port, our ability to interdict him as far away from shore as possible reduces the potential danger to the populace.

A common thread to all of the experiments, however, has been the need to establish network connectivity between the boarding party on board the suspect vessel and the rest of the TNT network (simulating GIG connectivity). Additionally, with a potentially large ship to search and a small boarding team, affording the boarding team connectivity among themselves was also important, allowing each member to reach all the way back to experts located on the other side of the country to help in the accomplishment of the mission.

3. Collaborative Network

The level of expertise required to successfully prosecute a MIO is, unfortunately, more than we can expect of the Navy and Coast Guard members who are actually trained for and tasked with the boarding activities. We can, and do, equip them with appropriate chem/bio and radiation detectors, but even with such advanced capabilities, they lack the skills necessary to interpret the information provided by their detectors.

Conversely, the scarcity of personnel able to interpret such data makes it impossible to simply make sure there is a chemical expert, a biological hazard expert, a radiation expert, a nuclear machine parts expert, etc. on every VBSS undertaken in the MIO environment. By linking these low-density, high-demand experts electronically to the operators engaged in the VBSS, we can significantly increase the level of expertise that can be leveraged against the problem and do it in a timeframe that allows for meaningful action to be taken. Another high-level network schematic follows as Figure 19. This Figure shows the various network nodes in the San Francisco Bay area or connected via VPN during a recent TNT-MIO experiment.

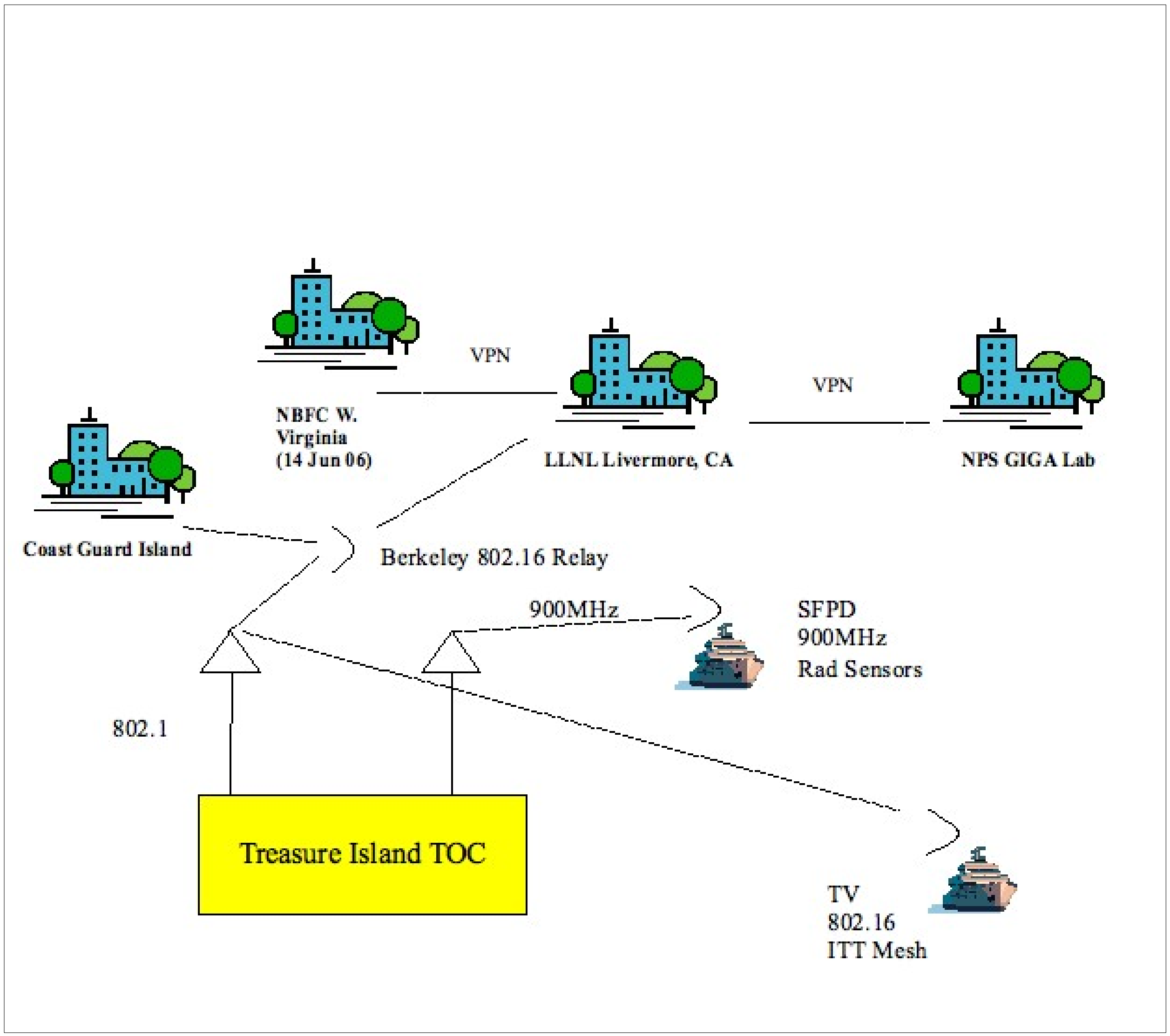


Figure 19. TNT-MIO Sites

In order to allow all the involved parties to communicate, each participant has access to the Microsoft Groove Virtual Office. This peer-to-peer computer mediated communication system allows users to communicate via threaded discussions, shared file spaces, chat and instant message. All parties involved in the MIO simulation were participants in one of several Groove workspaces. In this manner, the boarding party could take pictures of suspected nuclear proliferation materials and immediately make them available to the Defense Threat Reduction Agency (DTRA) agents in the experiment. The DTRA agents could then make a determination on whether the material was problematic or not or if more information was needed.

Groove is not a particularly sophisticated tool, nor is it designed for the kind of operations into which we have pressed it. However, the level of results that have been achieved in this makeshift fashion are significant enough to suggest that the concept is sound and that continued study is warranted. To understand the power created by linking remote experts with the boarding party in real or near-real time, consider the following: currently, if a boarding party suspects that a crew member or passenger on board a vessel may be a terrorist, they will take fingerprints and pictures of the suspect during the VBSS. If, however, they do not find anything to justify restraining the vessel or arresting the suspect, they will allow the vessel to continue.

Once back ashore or aboard their command vessel, they will submit the information they have in an attempt to identify the suspect. Response to such a request is on the order of hours or days, and that delay only begins after the boarding team has returned to the shore or their command vessel. Often, even if a positive match can be made, the suspect has long since fled, their ship having pulled into port hours or days before the boarding team had any actionable information.

With the use of collaborative technologies and adaptive ad-hoc networking, TNT-MIO experiments have shown the ability to return a positive match within 4 minutes of collecting biometric data. While this is under somewhat controlled experimental conditions, results even within an order of magnitude of this time allow the boarding party to take action while they are still on board the suspect vessel and long before the suspect can evade.

4. Hypernodes and TNT-MIO

As impressive as these results seem, we believe that there is room for improvement and that utilizing Hypernodes will allow us to realize those. For instance, under experimental conditions, the boarding party knows exactly which experts they need to contact in case they require assistance. If they were, instead, faced with a novel situation, how would they discover who to contact? If their network was populated by service aware Hypernodes, they could simply search for one that offered the required service. Finding a radiological source, they would look for a node providing radiation expertise.

Network management, too, is a significant problem during the MIO. The boarding party is unlikely to have, internally, the expertise required to manage and maintain their on-board network or the network link connecting them back to shore or their command vessel. Our current experimental setup attempts to overcome this limitation by utilizing a deployable NOC (DNOC); this gives the boarding party the tools, if not the expertise, to monitor and manage the network themselves. By utilizing subnetwork aware Hypernodes, a centrally located network manager can monitor and maintain even their remote network without severely impacting its performance by constantly polling. Here is an opportunity for improvement.

The decision space is the one where Hypernodes have the opportunity to make the most immediate and obvious impact. Most of the communication in the Groove workspace is centered on sharing or attempting to gain a shared awareness of the various decision states during the exercise. Has the commander decided there is a need for a more thorough search? Have the results come back from the experts? Do we have orders on how to proceed? Do the experts need more information from the boarding party? Has the boarding party given them the information they need? Capturing and sharing these decision states via Hypernodes can significantly reduce the amount of time and human communication necessary to establish a shared mental model in the collaborative workspace.

5. A Sample Scenario

Attached, as Appendix C, is a partial experimental plan for the TNT 06-4 MIO experiments from 29-31 August, 2006. While the names and positions of the specific actors have been removed, as well as many of the purely administrative details, it should serve to illustrate the kind of situations that are indicative of the MIO experiments. The objective of this scenario is:

> to continue to evaluate the use of networks, advanced sensors, and collaborative technology for rapid Maritime Interdiction Operations (MIO); specifically, the ability for a Boarding Party to rapidly set-up ship-to-ship communications that permit them to search for radiation and explosive sources while maintaining network connectivity with C2 organizations, and collaborating with remotely located sensor experts.
>
> The experiment extends the number of participating organizations beyond TNT 06-2 MIO to include three international teams in Sweden, Singapore and Austria, Oakland Police and Alameda County Marine Units in the MIO scenario. The networking elements of the experiment are also extended by innovative self-aligning broad band wireless solutions to support boarding and target vessels on-the-move.
>
> The experiment is expected to provide the necessary insight on transforming advanced networking and collaborative technology capabilities into new operational procedures for emerging network-centric MIOs.

The boarding party is faced with finding and identifying a radiation sample on board the target vessel as well as searching for any machine parts that could be a threat (either nuclear technology or bomb parts, etc.) They must also collect biometric data and identify any known terrorists among the crew. Additionally, several overseas participants have been added to increase the realism in the scenario. Information pertaining to the prior identification of some contraband material leaving Austria is available to the boarding party and the remote experts.

As mentioned before, there are some artificialities associated with the experiment, like the remote experts being known to the boarding party and standing by. Aside from that, however, the realism of the experiment is very high. The VBSS is even executed by an actual Coast Guard boarding team while the higher headquarters functions are played by Coast Guard District 11. Due to such realism, and the ability to log all communications in Groove, this provides an excellent source of raw communications data for analysis.

IX. EXPERIMENTAL METHOD AND RESULTS

A. EXPERIMENTAL METHOD

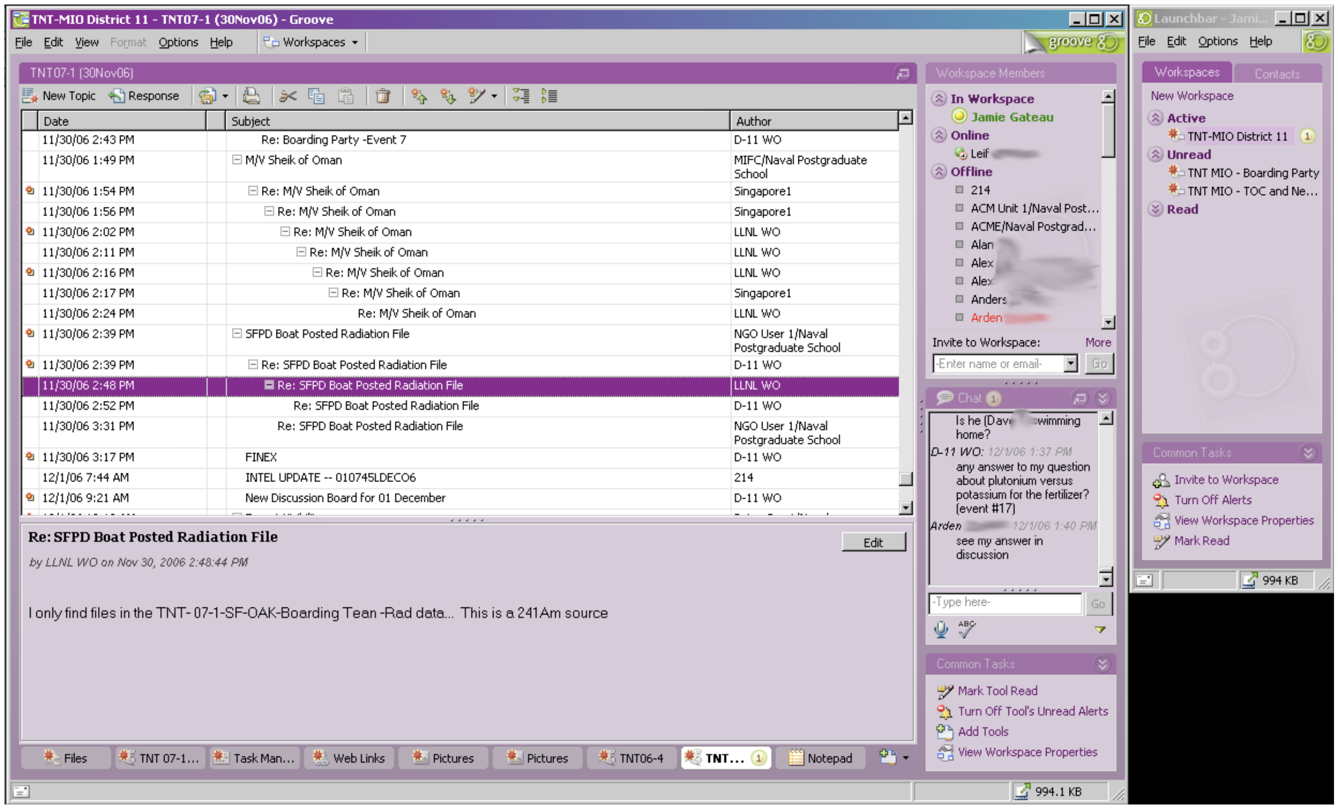
The use of Microsoft Groove Virtual Office as the primary means of communication during the TNT-MIO experiments enables us to capture and analyze, post-hoc, the transfer of information during the experiments. In order, then, to begin developing appropriate MIBs to express the service aware and decision support aware Hypernodes that could support future TNT-MIO experiments and the equivalent real-world operations, we take a qualitative look at this data.

Data are available from three TNT-MIO experiments: TNT 06-3, 06-4 and 07-1. These three experiments took place in June, September and December of 2006 respectively. The experiments are numbered using the federal government's fiscal year system, which starts in October, resulting in the seemingly odd nomenclature for the December experiment.

1. Groove Virtual Office

Groove allows for text-based communication using several tools: chat, discussion boards and instant messaging. Complete text logs are available for the chat and discussion boards from all three experiments, although instant message information is only partial. Because instant message information is purely peer-to-peer, no server-based logging exists of these messages. Where possible, the logs from each participant's software client were obtained, but this results in very spotty data capture.

During each experiment several different "workspaces" were in use. These workspaces (shown in the "launchbar" on the right hand side of Figure 20) were created to insulate the disparate communities of interest within the experiment. The Boarding Party had their own workspace for intragroup communication. The District 11 workspace included the majority of users, the outside experts as well as the Boarding Officer. The TOC and Networking workspace was used as an experiment control channel for the researchers and for network operations communications not related to the experiment.


TNT-MIO District 11 - TNT07-1 (30Nov06) - Groove
File Edit View Format Options Help Workspaces
groove
TNT07-1 (30Nov06)
New Topic Response

Date	Subject	Author
11/30/06 2:43 PM	Re: Boarding Party -Event 7	D-11 WO
11/30/06 1:49 PM	M/V Sheik of Oman	MIFC/Naval Postgraduate School
11/30/06 1:54 PM	Re: M/V Sheik of Oman	Singapore1
11/30/06 1:56 PM	Re: M/V Sheik of Oman	Singapore1
11/30/06 2:02 PM	Re: M/V Sheik of Oman	LLNL WO
11/30/06 2:11 PM	Re: M/V Sheik of Oman	LLNL WO
11/30/06 2:16 PM	Re: M/V Sheik of Oman	LLNL WO
11/30/06 2:17 PM	Re: M/V Sheik of Oman	Singapore1
11/30/06 2:24 PM	Re: M/V Sheik of Oman	LLNL WO
11/30/06 2:39 PM	SFPD Boat Posted Radiation File	NGO User 1/Naval Postgraduate School
11/30/06 2:39 PM	Re: SFPD Boat Posted Radiation File	D-11 WO
11/30/06 2:48 PM	Re: SFPD Boat Posted Radiation File	LLNL WO
11/30/06 2:52 PM	Re: SFPD Boat Posted Radiation File	D-11 WO
11/30/06 3:31 PM	Re: SFPD Boat Posted Radiation File	NGO User 1/Naval Postgraduate School
11/30/06 3:17 PM	FINEX	D-11 WO
12/1/06 7:44 AM	INTEL UPDATE -- 010745LDEC06	214
12/1/06 9:21 AM	New Discussion Board for 01 December	D-11 WO

Re: SFPD Boat Posted Radiation File

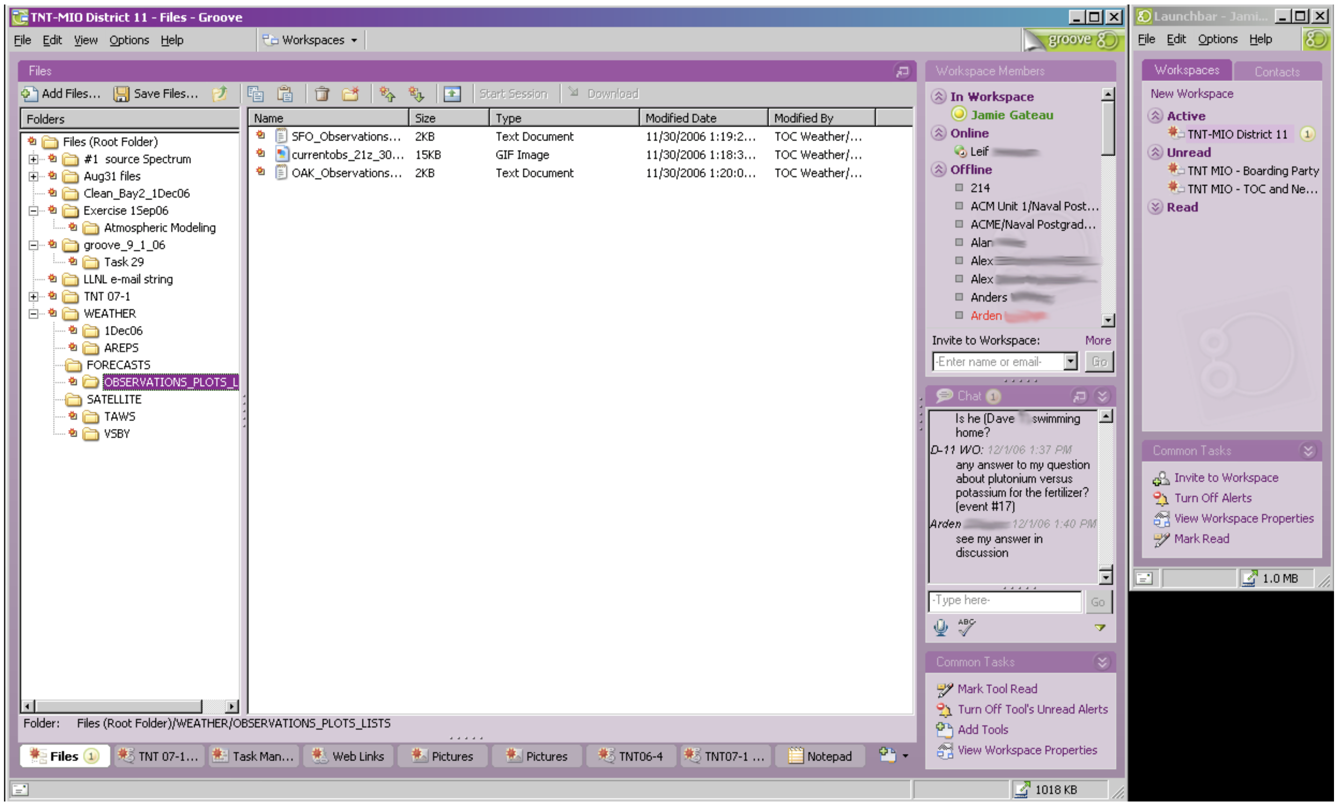
by LLNL WO on Nov 30, 2006 2:48:44 PM

I only find files in the TNT- 07-1-SF-OAK-Boarding Tean -Rad data... This is a 241Am source

Is he (Dave [redacted] swimming home?
D-11 WO: 12/1/06 1:37 PM
any answer to my question about plutonium versus potassium for the fertilizer? (event #17)
Arden [redacted] 12/1/06 1:40 PM
see my answer in discussion

Figure 20. Groove Workspace

The utilization of multiple workspaces was due primarily to a need to segregate the experimental control channel (TOC and Networking) from the experiment. A secondary desire was to insulate the Boarding Party's internal communications from the broader group. This is due to a limitation of the Groove software; the only means of access control is at the workspace level, therefore anyone with access to a workspace has access to all the information within that workspace. The Boarding Officer, in charge of the Boarding Party, was a member of both the Boarding Party workspace and the District 11 workspace, thereby allowing filtered communications between the workspaces.



Name	Size	Type	Modified Date	Modified By
SFO_Observations...	2KB	Text Document	11/30/2006 1:19:2...	TOC Weather/...
currentobs_21z_30...	15KB	GIF Image	11/30/2006 1:18:3...	TOC Weather/...
OAK_Observations...	2KB	Text Document	11/30/2006 1:20:0...	TOC Weather/...

Folder: Files (Root Folder)/WEATHER/OBSERVATIONS_PLOTS_LISTS

Is he (Dave [redacted] swimming home?

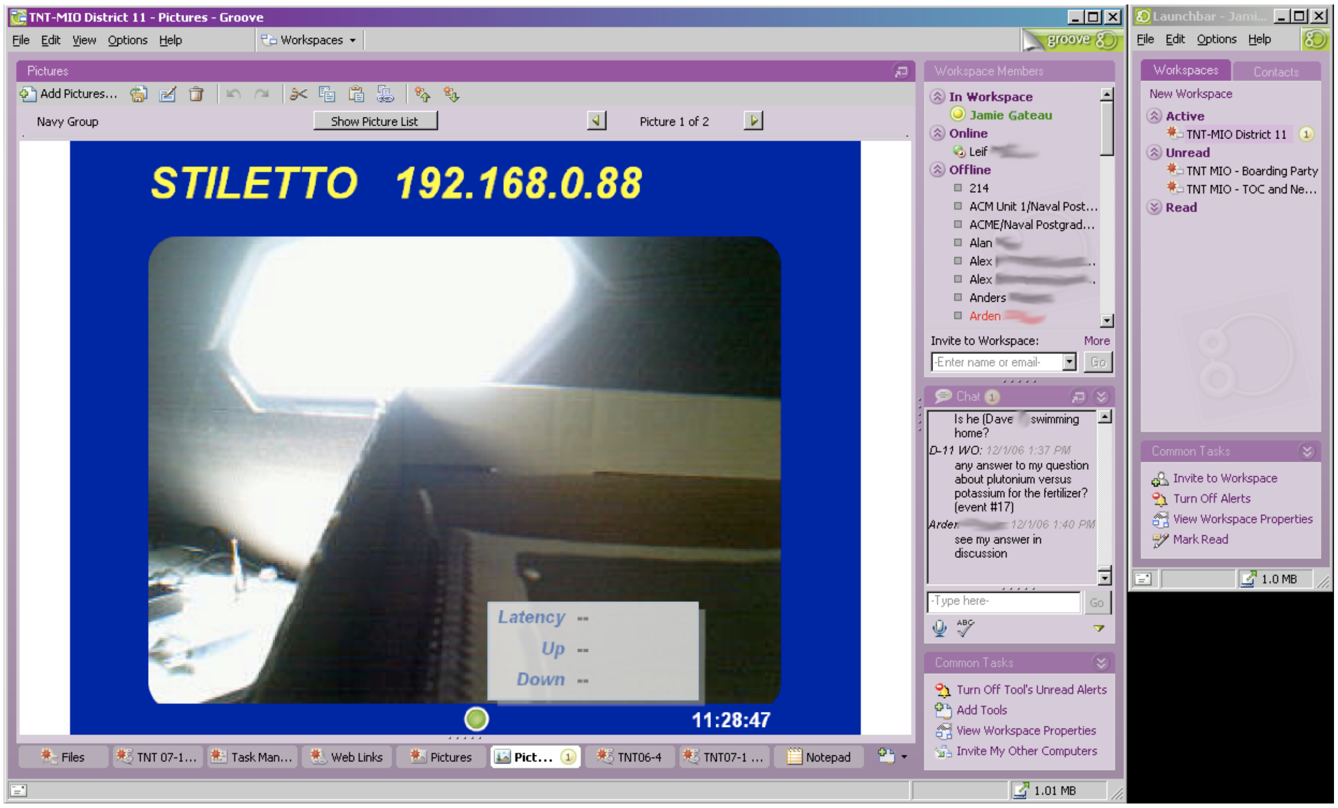
D-11 WO: 12/1/06 1:37 PM
any answer to my question about plutonium versus potassium for the fertilizer? (event #17)

Arden [redacted] 12/1/06 1:40 PM
see my answer in discussion

Figure 21. Groove File Sharing

Figure 20 is a screen shot of the Groove software running with the discussion board open in the main window. Discussions are threaded, such that responses to a message appear connected to the original message (see the multiple messages titled re:MV Sheik of Oman). When new messages come in, they are indicated with a red and yellow starburst (several are visible in Figure 20). Because of the threaded nature of these messages, new messages are often not located in temporally related locations; a starburst indication in the tab (bottom of Figure 20) indicates that there is a new message somewhere in the discussion, although it may require some looking around to find.

Figure 21, also a Groove screen shot, shows the file sharing space within the District 11 workspace. As in the previous screen shot, starbursts indicate new files or the existence of new files within a folder, as shown on the left. Figure 22 illustrates the picture utility in Groove. While pictures can be shared in the file area, the picture tool allows users to view the images directly in the Groove software instead of opening them in a separate application. In the normal course of use, both files and pictures are posted in the respective areas and a text message is sent either via chat or the discussion board to alert users.



Is he (Dave [redacted] swimming home?

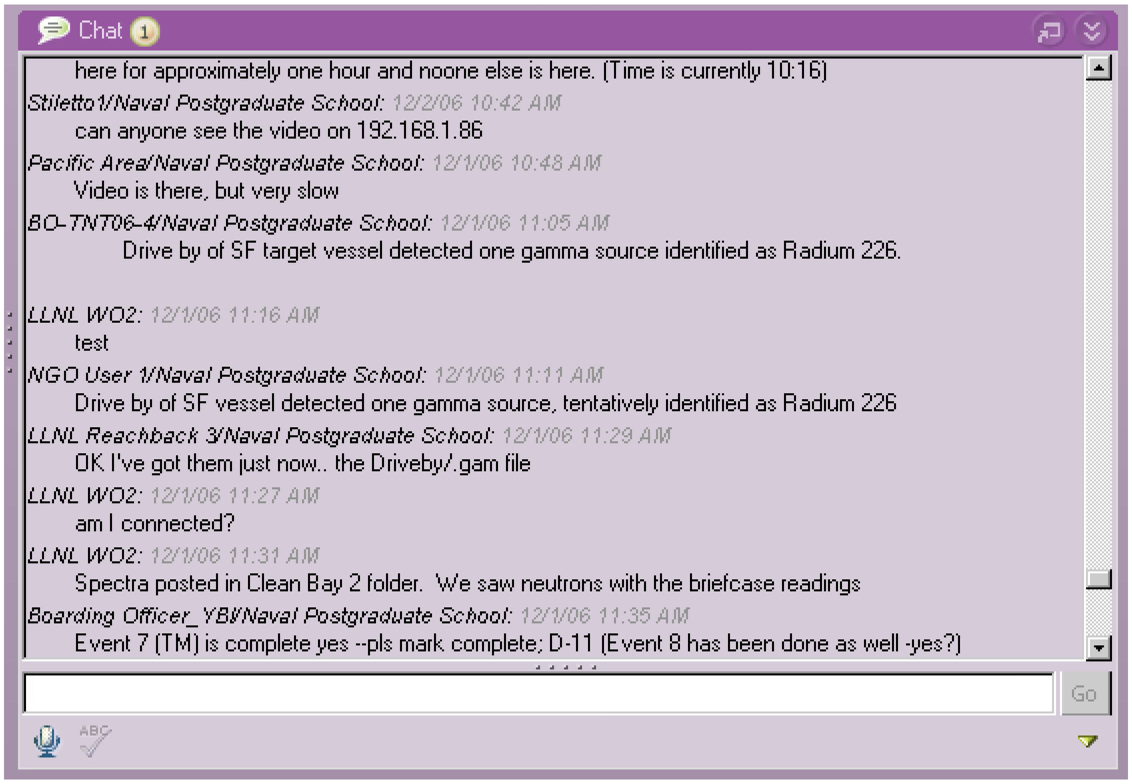
D-11 WO: 12/1/06 1:37 PM
any answer to my question about plutonium versus potassium for the fertilizer? (event #17)

Arden [redacted] 12/1/06 1:40 PM
see my answer in discussion

Figure 22. Groove Picture Tool

In all three figures, the chat window is visible on the right hand side of the screen below the user list. This chat tool provides a less structured way for members of the workspace to communicate than the discussion board. A magnified view of the chat tool is shown in Figure 23. This Figure also illustrates the ability of Groove users to resize the various tools to better fit their requirements. Here the chat tool has been expanded at the expense of other tools. Each message in the chat tool is arranged chronologically with the sender identified.

In addition to the starburst indications, Groove alerts users to new information via pop-up messages in the Windows status bar. Clicking on such a pop-up message takes the user directly to the new message wherever it may be in the Groove workspace. In a busy workspace, such as during the TNT-MIO experiments, it is not unusual for many participants to overlook the arrival of new information, even when it is something they are waiting for.



```
      here for approximately one hour and noone else is here. (Time is currently 10:16)
Stiletto1/Naval Postgraduate School: 12/2/06 10:42 AM
      can anyone see the video on 192.168.1.86
Pacific Area/Naval Postgraduate School: 12/1/06 10:48 AM
      Video is there, but very slow
BO-TNT06-4/Naval Postgraduate School: 12/1/06 11:05 AM
            Drive by of SF target vessel detected one gamma source identified as Radium 226.

LLNL WO2: 12/1/06 11:16 AM
      test
NGO User 1/Naval Postgraduate School: 12/1/06 11:11 AM
      Drive by of SF vessel detected one gamma source, tentatively identified as Radium 226
LLNL Reachback 3/Naval Postgraduate School: 12/1/06 11:29 AM
      OK I've got them just now.. the Driveby/.gam file
LLNL WO2: 12/1/06 11:27 AM
      am I connected?
LLNL WO2: 12/1/06 11:31 AM
      Spectra posted in Clean Bay 2 folder.  We saw neutrons with the briefcase readings
Boarding Officer_YBI/Naval Postgraduate School: 12/1/06 11:35 AM
      Event 7 (TM) is complete yes --pls mark complete; D-11 (Event 8 has been done as well -yes?)
```

Figure 23. Groove Chat Tool

In recent experiments, we have also tried using multiple discussion threads within a single workspace and relying upon users to contribute only to the correct discussions. This has led to less user confusion, but creates problems with, for instance, the chat logs, since all workspace chat occurs in a single channel. Due to the nature of data analysis undertaken for this thesis, none of these should present a detrimental effect to our results.

2. Data Analysis

These data are being treated as three related cases and are analyzed using the case study methodology as suggested in Yin (2003). Additionally, as the nature of communication is generally unconstrained in content, a constant comparison method was used to elicit categorical relationships among the various communiqués (Glazer and Strauss, 1967). Each message was analyzed individually with two foci of investigation: decision states and services.

Since the subnetwork aware Hypernode's MIB is unrelated to the context of the situation, being reflective more of the nature of networking in general and the specific hardware used, our analysis focused on communications which were indicative of the expression of services provided by the network and decision states within the network. The Groove logs for all three experiments were repeatedly inspected, looking for repeated themes. These themes were then collected to determine the nature of information sought by the various participants in the experiments.

When thematic analysis showed a recurring information need (such as querying for a decision state), these needs were considered as potential starting points for MIB development. As an example, from the TNT 06-4 experiment, at time 11:19, there was a request for a confirmation on a radiation detection event. At 11:35, another user was looking for the results of such a confirmation. The original text is as follows:[16]

```
------------------------------------------------------------------------
Request a maritime unit to confirm radiation detection
By Leif  on 8/31/06 11:19 AM
        Req MU for confirmation
------------------------------------------------------------------------
        Re: Request a maritime unit to confirm radiation detection
        By MIFC/Naval Postgraduate School on 8/31/06 11:35 AM
                For ALCO - have you performed the drive-by?  If so, have
                you posted the radiation files for LLNL reachback?
------------------------------------------------------------------------
```

To understand the format of the messages, note that the first line is the subject of the post. Responses begin with "Re:" and will be indented to increasing levels to show responses to responses, etc. The next line, which

[16] Throughout the next two sections, some minor edits have been made to the original logs, primarily to remove identifying information about the participants.

begins with "By" identifies the sender of the message. Any following information is the message itself. Because of the relative informality of the message format, some messages have a blank body, as all the information was passed in the subject line.

For coding purposes, the first message above (from Leif) is coded as a "request for services." Leif, in this case, knows that he requires a confirmation on the radiation detection and that a maritime unit is the appropriate element to take action. Had the request, instead been to identify a unit for this confirmation, it would have been coded as a "query for service." The obvious difference is that in one case, we are requesting known services from a known agent. In the other, either the specific service or the agent is unknown.

The reply to this message, from MIFC (Maritime Intelligence Fusion Center) is making two queries. In this case, however, they are coded as "queries for decision state." Sixteen minutes after the initial request for service, at least one participant still does not know the outcome of the activity. Thus, the two-part query is first to determine if the other user has taken an action and second to determine what the result of such an action has been.

A second example from the continuing exchange related to this "drive-by" confirmation follows:

Re: Drive By
By LLNL WO2 on 8/31/06 12:14 PM

any info on neutrons?

> Re: Drive By
>
> By MIFC/Naval Postgraduate School on 8/31/06 12:17 PM
>
> just talked to the boarding vessel - no info on neutrons - neutron detector broken

----

The first question is a "request for information (directed)," as they are looking for more information about the outcome of a previous action and are addressing the request to a specific agent. The second message, however, is a new theme for us, as it actually contains a "response." Specific information which was sought is being offered. This is really a two-part message however, as it both changes the decision state, from unknown to know ("decision announcement"), but also provides the actual response. (It is one thing to know that a decision has been made, quite another to know what the decision is.)

Thus, this qualitative analysis allows us to typify a number of different themes that recur throughout the three experiments. We will then use these themes to inform the creation of MIBs related to them. A treatment of a number of these recurring themes follows. It seems reasonable that future work to develop Hypernode MIBs will need to follow a similar pattern. Understanding the existing information flows within a phenomenon is a prerequisite for developing custom MIB variables if generic ones are insufficient. As this technology develops, of course, the number of exceptional phenomena (those requiring novel, non-standard MIBs) would be expected to decrease.

B. ANALYTICAL RESULTS

The following sections detail the primary themes discovered during analysis. Several of these will be treated further in Chapter X when we construct some candidate MIBs. Further analysis may, of course, result in more or different categorizations of the interactions that constituted the experimental transcripts. This is only one way of formalizing the interactions and may also not be exhaustive. We have attempted to categorize into as few themes as possible to express the entire range of interactions without losing nuance where such is valuable. Anyone interested in conducting a similar analysis is invited to request the raw log data.

1. Request for Service/Query for Service

Two recurring themes, mentioned above, are the Request for Service and the Query for Service. Requests were very frequent, as participants in this exercise were reasonably well briefed as to the capabilities of the other participants and to whom certain tasks were expected to be directed. It should be obvious, however, that this will not always be the case, and as such, it is postulated that a Query for Service will also be required. This was one of the most frequent exchanges that occurred during the experiments and took many forms.

"LLNL Watch Office: In addition, we need any atmospheric modeling data and any HOPS mapping," and "request drive by to confirm radiation alarm" are two examples of such requests. Another common request was for "reachback." This is a jargon term, understood by the team to mean that external expert assistance was required. They were not always phrased in such obvious ways, however. The statement, "I can't, but LLNL WO may be able to. If there's neutron activation, it's not foundry sand," is an implicit service request to the LLNL WO[17] to make a determination about a previous statement.

2. Request for Information (Directed)/(Undirected)

Related to the service requests and queries are the requests for information. These came in two distinct flavors: directed and undirected. A directed request for information addressed a specific party (sometimes implicitly) in order to gain some information (again in the Shannon sense (1949)) to reduce ambiguity. These themes occurred as frequently as did the service requests and also took varied forms, from explicit and well-structured, to implicit and ill-structured.

One expansive and well-structured example is,

> Got the data. Many questions: Where are the measurements taken? Specific sites? What types of detectors? Gamma? Neutron? Spectroscope? What do the different color curves represent? What was the object that set off the alarm - Cargo, vehicle, individual, etc. Is there an occupancy sensor? This would help me understand background levels better. Is the raw data available? Are there images available?

This message was directed to the agent who had sent a request for service and provided only partial information. These questions, while allowing some freedom in response, clearly delineate the kind of information that would represent an answer. To contrast, "What happened to the drive-by for event 6?"

[17] Lawrence Livermore National Laboratory Watch Officer.

is an ill-structured (and as it turns out, undirected) request for information. This is not a problem for the SNMP MIB or Hypernode constructs, it merely suggests that the agent looking to return an information response may need to ask for amplification or may return date which is not useful as information.

3. Response/Amplification

Obviously, if we are going to make queries and requests, there must be some kind of response. These responses are appropriately numerous in the data and take many forms, mimicking the varied requests that instigated them. A special kind of response also occurred. While the information content is no different from a normal response, an amplification modifies existing data, and, as such, will be treated differently in the MIB.

Amplifications can also be used to modify queries and responses. Since the basic mechanism remains the same—modification of existing MIB information, there is no difference between these two kinds of amplification in the Hypernode architecture. Thus, "Is this a sodium iodide detector?" (adding specificity to a previous question) and "Target visibility plot updated" (adding information to a previous answer) are equivalently treated.

4. Unsolicited Information

Several times, agents in the experiment offered unsolicited information. This may be in response to a sort of implicit information request that the agent understood to be in effect, or may result from an agent in possession of information that they suspect will be of value to other participants even though it has not be specifically requested. In the SNMP architecture, this is likely best addressed as a trap message, allowing the information source to provide the data to whomever they believe appropriate.

Several messages of the form, "Radiation Alert! File posted in folder Sweden." occurred throughout the experiment. These messages are of apparent importance to multiple actors, since they often resulted in further actions, but were not in response to any specific request. Also, in many cases, an information response to one actor may also be useful to others. Unsolicited information traps, then, allow both smart-push and smart-pull information architectures to be utilized, even both for a single piece of information.

5. Network Registration/Deregistration

Often, as agents came onto the network, they announced their presence. Also, as they left, they made similar announcements. This was often accomplished via the chat function of Groove and suggests these were of lower-priority, or perhaps less important to retain than discussion board messages. This registrations/deregistration keyed other network members to the availability of certain services on the network due to the presence of the member in question. The exchange, "I'm here." "So am I." "We are also here." typifies this kind of announcement.

As a MIB variable, these can be handled as simple registration/deregistration messages. In case of a hierarchical organization, as new members come on line, they simply register with the superior Hypernode. In the case of more Edge-like or flat organizations, network members may choose to register only with a subset of network members, or with all of them. This also suggests another utility for relationship awareness as a network service. Registrations/deregistrations could be propagated through the network as known relationships, allowing late registering nodes to gain awareness of the entire network without requiring the re-registration of all nodes.

6. Service Announcement/Status Announcement

Periodically, nodes found it necessary to announce the existence of new services, such as, "Video feed available at 83.209.68.158." These were generally broadcast-type, undirected announcements, but instead of pertaining to specific information, they reflected the availability of an ongoing capability. Above, we can see that the node with IP address 83.209.68.158 now had available a video feed. Without more information, however, we do not know of what this might be a video feed.

Fortunately, in this case, the announcement did include some metadata, "(onboard suspect vessel.)" A properly formatted service announcement will need to include enough metadata to be useful to other nodes receiving the announcement. This may be explicit in the service announcement or it may simply refer to some other MIB variables on the servicing node. If, in the previous case, the node had already been identified as being on board the suspect vessel, an announcement of "video feed available" might have been sufficient.

In the case where the availability of a service changed, status announcements were often made. These often took a form such as, "Sorry, that's the best we can do, the identiFiNDER is dead." Here, the actor had to respond to an information request by stating that the information was unavailable, and further that a service the node had previously provided (identification with the identiFiNDER) was no longer available. If such a service later returned, another service announcement would make such a notification to the network.

7. Decision Request

The decision states of the various network actors were of great interest to many other network members. "See ship matching intel description in San Diego Harbor requesting to conduct a drive by for radiation detection," served two information purposes.[18] First, it offered some unsolicited information (again, there may have been an implicit request for such information, but this was not a response). Second, it requested a decision from higher headquarters: permission to conduct a drive by. It should be clear that this is not a service request, as the service provider is, instead, looking for permission to go ahead.

This is categorized as a "decision request" instead of a request for permission because the underlying mechanism remains the same. Had higher headquarters already decided to execute a drive-by, this decision request would match against that MIB variable and return TRUE. The entity making the request is ultimately asking for the respondent to announce their decision state. There is no need for the decision request to come from a potential executor. It was not

[18] It is interesting that this entire message was sent as the subject line of the message with no amplifying information.

uncommon for actors to inquire about decisions made that only laterally affected them. This appeared to be in an effort to better develop awareness of the larger situation.

Other requests, such as, "This is Boarding Officer -based on plume, request guidance on moving the target vessel. Where do you recommend we move?" This one, also, appears to be an information request, but is actually a decision request. The Boarding Officer is looking for someone to decide and then share with him the decision about whether the ship should be moved. These requests are often found with either information responses/amplifications or with unsolicited information attached.

8. Decision Announcement/Task Assignment

Although a decision in response to a decision request is merely handled as a response message, in much the same way that actors often made unsolicited information announcements, they often also made decision announcements and status announcements. These are analogous in format, but differ in content from the unsolicited information announcements. In much the same manner, too, a single decision may result in a response to the requester and a decision announcement to one or more other actors in order to disseminate understanding to the entire network.

"I have tasked the boarding team with performing this data collection," is an excellent example. This actor had already tasked the boarding team, effectively passing the status on the data collection decision, but also decided to make the announcement to the remainder of the team. Decision announcements should also set a local MIB variable so that later decision requests immediately receive the decision status from the Hypernode without intervention. Task assignments are also a type of decision announcement as they may or may not result from a specific decision request.

9. SITREP

The boarding party, especially, heavily utilized a specialized information response called a SITREP (Situation Report). These, unlike unsolicited information, are specifically expected by higher headquarters. Like unsolicited information, though, they do not have a one-to-one relationship with an information request. One example is, "sitrep: #4 bio sample sent at 1328 - negative response at 1331 - Done with crew members bio samples - ALL NEGATIVE. Waiting for response on #1 and 2 radiation sources."

Such an information announcement allows all members of the network to maintain awareness of the ongoing situation without a need to positively query the boarding team. Only if more or amplifying information was required, did specific requests get made. Utilizing a SITREP construct in this network reduces the amount of information requests that need to occur. It is anticipated that other expected, implicit, pro forma information announcements will be required in other domains.

X. CRAFTING MIBS

The three candidate MIBs developed for the TNT-MIO scenario we will discuss in this chapter are included in their entirety as Appendix D. Only individual items of interest will be duplicated here in the interest of space and readability. These MIBs borrow heavily from the syntax and organization offered by RFC-1155 and RFC-1213, so similarities noticed between these and those are not accidental. It should also be noted that there is really only one way to format the data according to specification, the differences among all MIBs stem only from what data you choose to share and how it is organized.

An interesting problem was identified in the construction of these MIBs. Unfortunately, tables in MIBs cannot be nested. This means, ultimately, that table entries can only be leaf nodes and not other tables! Especially in the subnetwork aware Hypernode, this reduces some of the elegance of the solution, but does not result in a loss of capability. This does mean, for instance, that the childMibTable, discussed later, cannot be organized beyond a single layer, placing all MIB values for all child nodes in the same table.

As mentioned in Chapter VI, this does not mean that the data storage must occur in the same flat table. This MIB, especially, is a good example of a MIB which would benefit from the implementation of a relational database as the storage mechanism for the actual data values. As such, there is an opportunity to introduce another layer of abstraction and better organize the data being held by this Hypernode.

None of the MIBs in Appendix D and described below should be considered as exhaustive. They provide only a starting point for future research and examples of how the analysis described in Chapter IX can be applied to actual MIBs. They are, however, complete and should be implementable as-is. Figure 24 displays these MIBs as children of a yet-to-be-assigned NPS node on the enterprises branch of the Internet (1.3.6.1) tree. A service aware (service-hyper-mib), subnetwork aware (subnet-hyper-mib) and decision support aware (ds-hyper-mib) MIB are each described below.

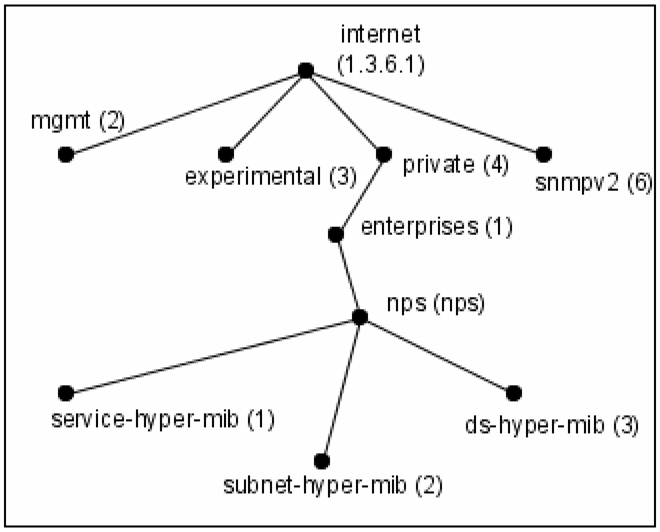


Figure 24. Hypernode MIB Tree

Aside from the three Hypernode MIBs discussed below, a fourth MIB is also offered in Appendix D, describing the highest level of the NPS tree. Currently, this contains only one variable of import to us, an INTEGER of nodeIsHypernode. If the node in question is a Hypernode, this value will be set using the same formula as childIsHypernode below. A zero value or the absence of this variable indicates that the node in question is not a Hypernode.

A. A SERVICE AWARE HYPERNODE MIB

The network service aware MIB provided in Appendix D is very generic in its implementation. It seems very likely that as further research is conducted, the details of the provided services table (proServTable) will call for a significant increase in the number of variables included. In its current form, this table allows only for querying to find the services provided by a given Hypernode. Any other interactions must occur via a different mechanism. The assumption made by the table is that a uniform resource locator (URL) can be utilized to access the service. This is understood to be a naive assumption at best.

Table 6 illustrates the entries in the proServTable in a more human readable format. Please see Appendix D for complete details.

Name	Syntax	Access	Description	OID
servIndex	INTEGER	read-only	A unique value for each service.	proServEntry 1
servName	DisplayString	read-only	A descriptive name of the service provided.	proServEntry 2
servReference	DisplayString	read-only	A unique reference for this service.	proServEntry 3
servDescr	DisplayString	read-only	A free-text description of this service. In the absence of other metadata, this description should be as complete as possible to allow other users to make decisions about the use of this service.	proServEntry 4
servIsMachine	BOOLEAN	read-only	TRUE if this service is automated.	proServEntry 5
servIsAvail	BOOLEAN	read-only	TRUE if this service is currently available. Additionally, a myServDown or myServUp trap should be sent to appropriate users when the Avail status changes.	proServEntry 6
servUrl	DisplayString	read-only	The Uniform Resource Locator where the service may be accessed.	proServEntry 7

Table 6. proServTable

This Table should be easily understood at this point. There are at least two comments worth making. The servIndex variable should remain as static as possible on a given system. Since the myServUp and myServDown traps reference the index, changes of indexes will cause confusion for other network users. The servIsMachine variable indicates whether this is an automated service or a human-provided service.

These two SNMP traps are also defined in this MIB. These allow announcements analogous to the service announcement/status announcement themes discovered in Chapter IX. When fired, these traps simply return the OID value for the index of the service in question. If, for instance, a service previously active fails, a myServDown trap is fired to interested parties. We should be concurrently setting the servIsAvail variable to FALSE if this is the case. Similarly, when the service returns to operation, the myServUp trap is fired. In both cases, the payload of the trap is the index of the affected service.

This is also useful if we are bringing a new service on line. When a new service is registered on the Hypernode, the generation of a myServUp trap serves to announce the new service to the network. The fact that we are not actually changing the status from down is immaterial. It may help, in fact, to think of it as changing the status from none to up to understand why this is the case.

Queries for service can be handled by requesting all the proServTables on the network and simply performing a simple search. This is particularly inelegant and we may wish to, instead, have a single Hypernode which does this and then offers "Service Search" as a service itself. This sort of construct will likely require the addition of another high-level npsMib variable so that the search for a "Service Search" node does not regress to the previous problem.

B. A SUBNETWORK AWARE HYPERNODE MIB

The subnetwork aware Hypernode MIB is brutally simple in its implementation, but that is a reflection of the purpose of such a MIB. The primary purposes of the subnetwork aware Hypernode are to enumerate the remaining nodes of the subnetwork and to cache MIB information for any of the nodes that request it. In this first incarnation, the information to be passed is also fairly rudimentary.

Name	Syntax	Access	Description	OID
childIndex	INTEGER	read-create	A unique value for each child. Its value ranges between 1 and the value of childNumber.	childEntry 1
childName	DisplayString	read-create	An administratively-assigned name for this managed node. By convention, this is the node's fully-qualified domain name.	childEntry 2
childIp	IpAddress	read-create	The IP address of this child on this subnet. In cases where there is more than one IP per subnet, the child node will determine which IP to advertise.	childEntry 3
childDescr	DisplayString	read-create	A textual description of the child. This value should include the full name and version identification of the system's hardware type, software operating-system, and networking software.	childEntry 4
childLocation	DisplayString	read-create	The physical location of this node (e.g., `telephone closet, 3rd floor').	childEntry 5
childCached	BOOLEAN	read-create	TRUE if we are caching MIB data for this node.	childEntry 6
childIsHypernode	INTEGER	read-create	A value which indicates the class of Hypernode this child represents.[19]	childEntry 7

Table 7. childTable

For basic information, there is a table of "child nodes," called childTable, and simple count of total children, childNumber. Because of the one-to-one relationship of children to table entries, this means that there are only as many rows in the childTable as the value of childNumber. Table 7 shows the entries in childTable and the salient features of each one. All MIB variables have a STATUS of MANDATORY, therefore, the STATUS information is excluded from Table 7.

Most of the information in this Table should be self-evident. Name, location and description information can be copied directly from the system MIB of the child node if desired. The childCached value reflects whether or not this Hypernode contains MIB information for this child node. Currently, this is only a Boolean value of TRUE or FALSE, meaning that we are either caching all MIB information for the child or none.

19 This integer value is based on a formula as given in the MIB and Appendix D.

The childIsHypernode entry allows us to identify whether the child node is, itself, also a Hypernode and what kind of Hypernode it is. The integer value of this variable identifies which of the three kinds of Hypernodes currently devised it serves as. If new classes of Hypernodes are identified, the formula which defines this value is easily extensible and is explained in Appendix D. Readers familiar with the Unix attrib command will recognize this formula immediately.

All values of this table are set as read-create with the understanding that child nodes, when registering with the Hypernode, will simply edit the appropriate MIB values in the table. Using read-create, instead of read-write, allows for new entries to be added to the table. It is expected that appropriate security measures will be implemented, either via SNMPv3 or some other means, to restrict access to these variables. If this is not possible, it will be sufficient to override the read-create value with read-only and make changes to this table on from the Hypernode via a programmatic method.

The other table defined in this MIB is called childMibTable, and contains the MIB values for all child nodes for which this Hypernode caches. As currently conceived, this is, quite literally, a table with every single MIB value on the subnetwork in it. As mentioned above, this may not be the most efficient way of actually storing the data, but it does make for a very simple representation. Table 8 describes the childMibTable.

This table represents a completely brute-force method of caching MIB values, but allows for an arbitrary nesting of cached nodes. Since the childMibTable is, itself, a number of MIB values, if one subnetwork aware Hypernode is caching for another each of the child MIB values is also a Hypernode MIB value. If the Hypernode is also cached at a higher level, all the child MIB values of which it is aware will also be passed up to the caching Hypernode.

Name	**Syntax**	**Access**	**Description**	**OID**
childMibIndex	INTEGER	read-create	A unique value for each cached MIB entry.	childMibEntry 1
childIp	IpAddress	read-create	The IP address of this child on this subnet. In cases where there is more than one IP per subnet, the child node will determine which IP to advertise.	childMibEntry 2
childMibOid	DisplayString	read-create	The OID being cached.	childMibEntry 3
childMibValue	DisplayString	read-create	A copy of the MIB value from the child node. Since we can't be sure of the syntax for the cached information, we use a display string.	childMibEntry 4
childMibDescr	DisplayString	read-create	A copy of the MIB description from the child node. This creates overhead in the case that this is a standard MIB value, but allows us to be self-describing even for previously unknown OIDs.	childMibEntry 5

Table 8. childMibTable

This construct also creates a one-to-one relationship between Hypernode MIB values and child MIB values. For instance, if the System Description MIB variable of a cached node is located in 1.3.6.1.4.1.nps.2.1.3.1.1, retrieving 1.3.6.1.4.1.nps.2.1.3.1.1.4 on the Hypernode returns the same information as retrieving 1.3.6.1.2.1.1.1 on the cached node. Thus, we no longer must query end nodes directly to return their MIB information.

C. A DECISION SUPPORT AWARE HYPERNODE MIB

The decision support aware Hypernode MIB takes much the same form as the service aware MIB. This is not surprising when we consider that decision support could have been implemented as a specific kind of service provided by the Hypernode. Such a decision may be useful in the future, but we have chosen to differentiate them at this stage because they fill significantly different spaces in the TNT-MIO domain.

Table 9 details the basic decision state table. This table, madeDecisionTable, has a slightly misleading title. While this table does contain finalized or made decisions, it also contains new decisions and those in progress. As it is expected that closed decisions will remain in the table for much longer than the decision-making process itself, we expect that the majority of entries in this table will have been previously made decisions at most times.

In keeping with the previous MIBs, most of the information should not require explanation. Two comments are noteworthy, however. The decisionIndex variable should remain unchanged for at least the lifecycle of a decision. If, for instance, a given node makes decisions with a three-day timeline (an ATO cycle, for instance), index numbers should not be recycled inside this frequency. Since the upper bound on integers is extremely large in human terms (232) it will likely be worthwhile to never recycle decision numbers until the counter "rolls over." It is expected that maintaining unique decisionNames is likely to pose more of a problem than running out of index space.

Name	**Syntax**	**Access**	**Description**	**OID**
decsionIndex	INTEGER	read-only	A unique value for each decision.	madeDecisionEntry 1
decisionName	DisplayString	read-only	A descriptive name of the decision. This should be a short but unique identifier of the decision.	madeDecisionEntry 2
decisionDescr	DisplayString	read-only	A free-text description of this decision. In the absence of other metadata, this description should be as complete as possible to allow other users to take action based on this description.	madeDecisionEntry 3
decisionIsOpen	INTEGER	read-only	other(0), new(1), open(2), awaiting-amplification(3), closed-undecided(4), closed-decided(5)	madeDecisionEntry 4

Table 9. madeDecisionTable

The decisionIsOpen variable is an INTEGER, but takes an enumerated list as possible values. New decisions, those on which the decision maker has taken no action toward a decision, are classified as 1. Once the decision maker or his delegate has begun to work on a decision, it should be set to 2, open. If the decision maker requires amplification or information and therefore cannot make a decision, the value should be set to 3.

If this decision maker chooses to transfer, abdicate or simply table a decision without a resolution, it should be set to 4 so that other network members know not to expect any further change on this decision from this node. A closed-decided decision (5) indicates that a decision has been made, with appropriate description information in decisionDescr. The value of 0 (other) should be used when no other value is appropriate. In this case, amplifying information should be placed in decisionDescr.

The existence of the madeDecisionTable allows other network members to issue SNMP get requests in order to determine a decision state without needing to directly contact anyone responsible for the decision. This alone would have significantly reduced the amount of discussion board and chat traffic for the TNT-MIO experiment, and when coupled with the ability to issue a trap when decision states changed, could reduce the manual information flow to purely exceptional issues.

This MIB also defines two traps, the second of which applies to this table. The myDecisionChangeState trap is issued any time the decision maker changes the decisionIsOpen variable, indicating to network members that the decision whose index is in the trap has changed state. Unfortunately, due to the nature of the SNMP trap, that single value is all that is available. Interested parties must still then execute an SNMP get to determine the new value of the variable.

Finally, Table 10 describes a vehicle for allowing nodes to ask for decisions. The ds-hyper-MIB also implements a table to support such requests called dsInboxTable. Where the variables in the prior table were all read-only, these are all available to be modified or created by other nodes. It is expected that a separate mechanism will be used to transfer these submitted decision requests into the madeDecisionTable.

The primary difference between this table and Table 9, aside from name changes is the introduction of the dsRequester value. This value contains the IP address of the node requesting the decision.

The remaining trap defined by this MIB, myInboxAccepted, simply notifies the original requester of a decision that it has been accepted for action by the decision maker. This message includes the inboxIndex of the submitted request. After creating the entry in the madeDecisionTable for the new entry, the decision maker's agent should also send a myDecisionChangeState notification, since the creation of a new madeDecisionEntry, necessarily, involves a change in state—from none to, most probably, new or open. In this way, the requester knows both that his decision has been accepted and what the OID is for the index to the decision so that they might monitor it later.

Name	Syntax	Access	Description	OID
inboxIndex	INTEGER	read-create	A unique value for each decision.	dsInboxEntry 1
dsInboxName	DisplayString	read-create	A descriptive name of the decision. This should be a short but unique identifier of the decision requested.	dsInboxEntry 2
dsInboxDescr	DisplayString	read-create	A free-text description of this decision. In the absence of other metadata, this description should be as complete as possible to allow the decision make to take this for action.	dsInboxEntry 3
dsRequester	IpAddress	read-create	The IP address of the node requesting the decision.	dsInboxEntry 4

Table 10. dsInboxTable

These three candidate MIBs go a long way to fulfilling the needs identified during the thematic analysis of the TNT-MIO data. They should still not be seen as a complete solution to a Hypernode implementation, even within this limited domain. They are, however, intended as both a proof of concept and template by which others may improve the capabilities of the Hypernode architecture, and even in this rough form, they offer significant capabilities to address identified requirements within the TNT-MIO experiments.

XI. EXTENSIONS AND FURTHER RESEARCH

Throughout this thesis, we have tried to develop an architecture for information exchange based on the Simple Network Management Protocol (SNMP). We first reviewed the context of Network Centric Warfare provided by the Global Information Grid and FORCEnet concepts. These concepts, along with the O-I-T multi-level model inform the direction we have taken with that architecture, attempting to address the information exchange problem at multiple levels in a service-oriented fashion.

After discussing the technologies which underlie our architecture, namely SNMP and ASN.1, we developed three classes of Hypernodes which we propose to address the current deficiencies of information exchange. These three classes were further explored with case study investigation of the TNT-MIO experiment. From this analysis, candidate MIBs were developed and explained as an initial step in the realization of this architecture.

In order to appropriately develop the theoretical background for a work of this size, however, it was necessary to sacrifice a number of the products originally planned for this thesis. What remains, we contend, will provide not only a thorough treatment of the subject matter as it pertains to the TNT-MIO experimental domain, but will also serve as an invaluable starting point for significant future research. A number of the originally planned products are obvious candidates for future work.

A. WEB SERVICES

Throughout this thesis, we have discussed SNMP as the primary means we wish to utilize to transfer data. This decision was made from a twofold desire to minimize the number of information formats in use on the network and to ensure that even disadvantaged nodes have the ability to participate with or as Hypernodes. One direction of future research should include the investigation of using Web Services in conjunction with or as a wrapper for the SNMP data.

At least one architecture is suggested wherein a number of SNMP Hypernodes also serve as Web Services gateways. These gateways would allow for translation from SNMP to, for instance, the WS-Management framework and back, allowing SNMP-enabled nodes to access information on non SNMP nodes and vice versa.

B. SNMP-SNMP GATEWAY

The subnetwork aware Hypernode offers a quantum leap in capability for management of large, distributed networks in a purely SNMP fashion. In the TNT network, alone, this could reduce by a factor of 2-3 the amount of network traffic consumed by management information on the fringes of the network where capacity is the smallest already. The tools currently in use by CENETIX do not, however, have an ability to directly monitor custom MIB variables. In our experience, this is a significant limitation on nearly all the toolsets available in the network management space.

An obvious extension of this work, then, would be the creation of an SNMP-SNMP gateway serving much the same function as the Web Services gateway above. In this manner, a normal SNMP request to a cached node would be intercepted by its Hypernode which would answer for it. Unfortunately, without such capability, non-Hypernode aware network management tools will not be able to take advantage of this advance.

C. COMPILED MIBS

The process for compiling the ASN.1 format MIBs, as given in Appendix D, into executable computer code turned out to be more complicated than originally expected, possibly equal in scope to an entire thesis itself. An obvious extension to this research would translate the ASN.1 MIBs into C/C++ code and compile them into a usable format.

D. USER AND MACHINE INTERFACES

Even with usable custom MIBs, the manual effort required to issue SNMP commands for the custom variables would overwhelm any positive effects of the Hypernodes themselves. A possible parallel activity to the MIB compilation would be development of user applications for interfacing with Hypernodes.

These would, necessarily, both include user-centric and machine-centric applications. A user-centric application might allow a user to query for services using a web form and return nodes offering matching or similar services.

Machine-centric applications would also be required. As mentioned in Chapter X, a number of changes to MIB variables should result in traps being sent throughout the network. This requires a helper application to watch the change of one variable in order to send such traps automatically. Database connectors also must be developed such that SNMP requests can be answered from robust data sources.

E. TESTING WITHIN THE TNT-MIO DOMAIN

Once the prerequisite MIBs have been compiled and deployed into the TNT network, testing should occur to determine both the usability and efficacy of the MIBs. The extent to which applications have been developed for this use will likely determine the means and method of testing. Involvement of users early in the program may also be useful for informing the development of the user interfaces themselves and for capturing previously unseen requirements.

F. OTHER DOMAINS/OTHER MIBS

It should be clear that the methods here are not designed purely for the TNT-MIO domain. It is our hope, in fact, that this architecture is appropriately flexible and powerful to be used as the information infrastructure in a wide range of domains. Consequently, future research designed to expand the applicability of the Hypernode concept to other domains is welcomed and encouraged.

There are likely other MIBs or extensions to the proposed MIBs both within the TNT-MIO domain and within others that will need to be developed. Hopefully, the method proposed by this thesis can also guide such work.

G. OTHER DECISION MODELS/DATA MINING

In early drafts of this thesis, it was pointed out that the decision information captured in a decision support aware Hypernode may be useful as a data source for later study. The results of and processes used for these decisions could inform later decisions. As currently structured, the decision support aware Hypernode MIB appropriately supports a case-based decision-making methodology.

Further research into the specific applicability of this decision information to later users should be investigated from the point of view of multiple decision-making methodologies. With this information, the decision support aware MIB should be modified or extended to give this capability to users of this class of Hypernode.

H. OTHER CLASSES OF HYPERNODES?

This thesis proposes three classes of Hypernodes, each filling a different requirement in the information space. These classes were chosen due to their applicability to the GIG and FORCEnet concepts. They were designed in order to maximize the number of levels in the O-I-T framework to which the made contributions. It is unlikely, however, that we have succeeded in exhausting the classes of Hypernodes available for exploration. It may even turn out that the Decision Support Aware Hypernode is, in fact, merely a special kind of service, collapsing the proposed three Hypernode types to only two. We welcome such criticism and invite future research along these lines.

APPENDIX A. FORCENET

A. FORCENET CAPABILITIES

(Reproduced from Clark and Hagee, 2005)

- Provide robust, reliable communication to all nodes, based on the varying information requirements and capabilities of those nodes.
- Provide reliable, accurate and timely location, identity and status information on all friendly forces, units, activities and entities/individuals.
- Provide reliable, accurate and timely location, identification, tracking and engagement information on environmental, neutral and hostile elements, activities, events, sites, platforms, and individuals.
- Store, catalogue and retrieve all information produced by any node on the network in a comprehensive, standard repository so that the information is readily accessible to all nodes and compatible with the forms required by any nodes, within security restrictions.
- Process, sort, analyze, evaluate, and synthesize large amounts of disparate information while still providing direct access to raw data as required.
- Provide each decision maker the ability to depict situational information in a tailorable, user-defined, shareable, primarily visual representation.
- Provide distributed groups of decision makers the ability to cooperate in the performance of common command and control activities by means of a collaborative work environment.
- Automate certain lower-order command and control sub-processes and to use intelligent agents and automated decision aids to assist people in performing higher-order sub- processes, such as gaining situational awareness and devising concepts of operations.
- Provide information assurance.
- ●Function in multiple security domains and multiple security levels within a domain and manage access dynamically.
- Interoperate with command and control systems of very different type and level of sophistication.
- Allow individual nodes to function while temporarily disconnected from the network.

- Automatically and adaptively monitor and manage the functioning of the command and control system to ensure effective and efficient operation and to diagnose problems and make repairs as needed.
- Incorporate new capabilities into the system quickly without causing undue disruption to the performance of the system.
- Provide decision makers the ability to make and implement good decisions quickly under conditions of uncertainty, friction, time, pressure, and other stresses.

APPENDIX B. RFC-1155 MIB

A. RFC-1155 MIB

The RFC-1155 MIB is included in this appendix for two reasons. One, the format of RFC-1155 is very straightforward and easy to understand, allowing readers of this thesis a complete example to examine in case any of the information, especially in chapters VI and VII, is too hard to understand in the abstract. Additionally, the MIBs developed as part of this thesis, as well as nearly all other network management MIBs currently in use, draw heavily upon the values exported by the RFC-1155 MIB. See Rose and McCloghrie (1990) for the entirety of the RFC and for any further explanation.

```
     RFC1155-SMI DEFINITIONS ::= BEGIN

     EXPORTS -- EVERYTHING
             internet, directory, mgmt,
             experimental, private, enterprises,
             OBJECT-TYPE, ObjectName, ObjectSyntax,
      SimpleSyntax,
             ApplicationSyntax, NetworkAddress, IpAddress,
             Counter, Gauge, TimeTicks, Opaque;

     -- the path to the root

     internet          OBJECT  IDENTIFIER  ::=  {  iso  org(3)
dod(6) 1 }

     directory     OBJECT IDENTIFIER ::= { internet 1 }

     mgmt          OBJECT IDENTIFIER ::= { internet 2 }

     experimental  OBJECT IDENTIFIER ::= { internet 3 }

     private       OBJECT IDENTIFIER ::= { internet 4 }
     enterprises   OBJECT IDENTIFIER ::= { private 1 }

     -- definition of object types

     OBJECT-TYPE MACRO ::=
     BEGIN
     TYPE NOTATION ::= "SYNTAX" type (TYPE ObjectSyntax)
                       "ACCESS" Access
```

```
                   "STATUS" Status
VALUE NOTATION ::= value (VALUE ObjectName)

Access ::= "read-only"
                | "read-write"
                | "write-only"
                | "not-accessible"
Status ::= "mandatory"
                | "optional"
                | "obsolete"
END

-- names of objects in the MIB

ObjectName ::=
    OBJECT IDENTIFIER

-- syntax of objects in the MIB

ObjectSyntax ::=
    CHOICE {
        simple
            SimpleSyntax,

-- note that simple SEQUENCEs are not directly
-- mentioned here to keep things simple (i.e.,
-- prevent mis-use).  However, application-wide
-- types which are IMPLICITly encoded simple
-- SEQUENCEs may appear in the following CHOICE

        application-wide
                ApplicationSyntax
    }

   SimpleSyntax ::=
       CHOICE {
           number
               INTEGER,

           string
               OCTET STRING,

           object
               OBJECT IDENTIFIER,

           empty
               NULL
```

```
     }

 ApplicationSyntax ::=
    CHOICE {
        address
            NetworkAddress,

        counter
            Counter,

        gauge
            Gauge,

        ticks
            TimeTicks,

        arbitrary
            Opaque

-- other application-wide types, as they are
-- defined, will be added here
     }

-- application-wide types

NetworkAddress ::=
    CHOICE {
        internet
             IpAddress
    }

IpAddress ::=
    [APPLICATION 0]        -- in network-byte order
         IMPLICIT OCTET STRING (SIZE (4))

Counter ::=
    [APPLICATION 1]
         IMPLICIT INTEGER (0..4294967295)

Gauge ::=
    [APPLICATION 2]
        IMPLICIT INTEGER (0..4294967295)

TimeTicks ::=
    [APPLICATION 3]
        IMPLICIT INTEGER (0..4294967295)
```

```
Opaque ::=
    [APPLICATION 4]        -- arbitrary ASN.1 value,
        IMPLICIT OCTET STRING   -- "double-wrapped"

END
```

APPENDIX C. FIELD EXPERIMENT

Joint USSOCOM-NPS-LLNL Field Experiment Augmented by OSD/HD MDA Programs TNT 06-04 San Francisco Bay, 29-31 August 2006

Objective:

The objective of this experiment is to continue to evaluate the use of networks, advanced sensors, and collaborative technology for rapid Maritime Interdiction Operations (MIO); specifically, the ability for a Boarding Party to rapidly set-up ship-to-ship communications that permit them to search for radiation and explosive sources while maintaining network connectivity with C2 organizations, and collaborating with remotely located sensor experts.

The experiment extends the number of participating organizations beyond TNT 06-2 MIO to include three international teams in Sweden, Singapore and Austria, Oakland Police and Alameda County Marine Units in the MIO scenario. The networking elements of the experiment are also extended by innovative self-aligning broad band wireless solutions to support boarding and target vessels on-the-move.

The experiment is expected to provide the necessary insight on transforming advanced networking and collaborative technology capabilities into new operational procedures for emerging network-centric MIOs.

1. PARTICIPATING UNITS AND ROLE PLAYERS

Participating Units

- **Alameda County Sheriff's Office Marine Patrol Unit Boat** and RHIB– Boarding vessel, deploys boarding party and does drive by (carries IST detector)
- **Oakland Police Boat 35** the target vessel
- **OFT Stiletto Ship-**remote early warning command post en route to San Diego area
- **USCG**
  - District 11 Watch Officer
  - PAC Area Watch Officer
  - MSST Level Two capable boarding team with radiation detection equipment?
- **LLNL**
  - Providing source, source security, and data files for detection

    teams (if necessary)
  - Providing remote analysis cell from Livermore via Groove
  - Provide mapping facility of bay showing critical facilities (HOPS), radiation detection reachback and atmospheric modeling reachback
  - LLNL Watch Officer – remote cell (operating from NPS)
  - 2 members of Boarding Party (with radiation detectors)

- **Innovative Survivability Technologies** Radiation detection software and ARAM detector for fixed sensor (on ALCO cutter)
- **BFC** (Biometrics Fusion Center)
  - Providing data files for detection teams,
  - Providing remote support for exercise database search and results reporting via Groove collaborative software
- **SOCOM** Observers
- **NPS**
  - Class on Collaborative Technologies
  - Network Operations Center and Data Collection site via groove
  - Network Support team and Experiment Control (act as back up to make all
    necessary inject should network connectivity problems exclude certain players).
- **Swedish Team**
  - Maritime Security Office of the Port of Oakland
  - observing and supporting experiment control by scenario injects made via groove, SA, and by video feed (with CDR Leif Hansson in Lead)
- **Austrian Team**
  - Port of Hong Kong (where the containers were loaded)
  - observing and supporting experiment control by scenario injects made via Groove, SA, and by video feed (with Dr. Ulrich Hofmann in Lead, Ulrich Wagner as Technical POC)
- **Team in Singapore**
  - Shipper of the cargo containers
  - observing and supporting experiment control by scenario injects made via Groove, SA, and by video feed (with Dr. Yu Chiann in Lead)
- **DHS Science & Technologies CounterMeasures Test Beds**
  - Office of Emergency Services
    - Assists CalOES and DOE RAP
- **California Office of Emergency Services**
  - Co-lead, Office of Emergency Services
- **DOE RAP**
  - Co-lead, Office of Emergency Services

2. ROLES and RESPONSIBILITIES

USSOCOM- NPS Experiment Director
MIO Experiment Coordinator and NPS PI
LLNL Coordinator

At Sea Participants:

Boarding Officers
Boarding Team Officer
Biometrics Collection
Radiation Detection – Level II inspectors (Boarding)
Deploys Boarding Party
Alameda County Sheriff's Boat
Target Ship, Oakland Police Boat #35
Target Ship Crew
Data Collection

Advisors and Geographically Distributed Experts:

USCG District 11 Watch Officer
USCG PAC AREA
LLNL Watch Officer

LLNL reachback - Radiological Analysis
LLNL Atmospheric Modeling
LLNL HOPS
(Homeland Defense Operational Planning System)
Biometrics Analysts
USSOCOM Observer
Experiment Scenario Control, SA, and Collaborative Environment
802.16 TNT MIO Backbone, Boarding Party mobile network, and SA software
Groove collaborative workspaces and VPN with partners

NPS NOC manager network support officers
Locations TBD
Swedish Team
Austrian Team
Team in Singapore
Human Systems – data collection
DOE RAP
Cal OES
DHS Science & Technology CounterMeasures Test Beds

3. DISCOVERY AND DEMONSTRATION TECHNOLOGIES

1. OFDM 802.16 Backbone Extension to SF Bay (alternative tactical paths):
    - Coast Guard HQ-Chabot Center-MARAD GEM STATE-ex-USS HORNET-Boarding Party-Target Ship
    - Coast Guard HQ-Lawrence Berkeley National Lab-Yerba Buena Island-Boarding Party-Target Ship
    - Stilletto
2. Remotely located GATE wireless surveillance network (Bavarian Alps) addition to MIO testbed (connected by TNT VPN, includes video and radiation detection sensors)
3. Remotely located Southern Sweden Boarding Party site addition to MIO testbed (provided by NPS VPN solutions, includes video sensors)
4. Mobile man-portable OFDM/802.16 network extension from TNT test-bed to Alameda County Sheriff's boat, SFPD Marine Unit boat and USCG MSST small boat Cutter
5. ITT Mesh on board of Alameda County Sheriff's Boat
6. Biometrics gathering device VPN reach back to various TNT-MIO collaborative partners.
7. Portable radiological detection devices. USCG: Identifinder (set to USCG specs), RadPager, SFPD: Sodium Iodide, IST: ARAM
8. Groove peer-to-peer Collaborative tool and Situational Awareness Agents
9. E-wall data fusion and situational awareness memory mechanism
10. VPN reach back to various TNT-MIO collaborative partners.

4. SCENARIO

4.1 Preamble

A truck loaded with a cargo container entering the port of Hong Kong sets off a portal monitor at the port. The Hong Kong Border Guards inspect the truck and find nothing unusual. The truck proceeds and the container is loaded on to a vessel owned and operated by a shipping company from Singapore. Later in a routine check of their instruments the Border Guards discover that they have set the background level too high on their handheld instruments.

A ship transiting into the San Francisco Bay (see map) creates an anomalous signature on a fixed radiation detector near Yerba Buena Island. The ship is headed to the Port of Oakland.

4.2 Major MIO Events

1. Preamble: A truck loaded with a cargo container entering the port of Hong Kong sets off a portal monitor at the port. The Hong Kong Border Guards inspect the truck and find nothing unusual. The truck proceeds and the container is loaded on to a vessel owned and operated by a shipping

company from Singapore. Later in a routine check of their instruments the Border Guards discover that they have set the background level too high on their handheld instruments (data sent to GROOVE [concurrently reflected in E-wall]).

   a. INJECT: Radiation alert is posted in District-11 workspace (from Hong Kong played by Austria)
   b. Hong Kong opens video feed to surveillance (V-Stream or Video-conferencing application)

2. Preamble: A ship transiting into the San Francisco Bay creates an anomalous signature on a fixed radiation detector near Yerba Buena Island. The ship is headed to the Port of Oakland.
   a. INJECT: Radiation alert is posted in District-11 workspace (from Port of Oakland played by Swedes)
   b. in EWALL, data sent to GROOVE)District-11, YBI and Boarding Vessel utilize SA Agent to see/track ship.

3. The Maritime Security Office (Swedes) request a maritime unit to do a drive-by of the ship to confirm the signal.

   a. ALCO MU(sends data to the Maritime Security Office of the Port of Oakland (played by Swedish group)
   b.
   c. Port of Oakland (Swedes) send radiation data to LLNL for radiation reachback.

4. INJECT: The Local Maritime Security Office (Swedes) reports the name and location of the ship, its registry and last port of departure to USCG and asks for guidance. (it is enroute to Oakland)
   a. D11/USCG asks the last port of departure for information regarding the ship

5. The US Coast Guard prepares to send a team to board the ship (USCG in lead now, notifies Boarding Party).
   a. Exercising the collaborative security agreement between Alameda County and Port of Oakland, Alameda County is notified of the issue via the collaborative network and the Maritime Security team on the Boarding Vessel enters the Groove workspace to coordinate the resolution of the issue.
   b. NPS Boarding Vessel Liaison to coordinate.
   c. The Maritime Security Office in Alameda County sends a level 2 boarding team to interdict, board and search the ship. Estimated time of arrival of the boarding team is 30 minutes.

6. The USCG informs the Port of Oakland that the MSO will no longer be needed.
7. Preliminary information from LLNL radiation reachback identifies two sources; one may be uranium.

8. The Coast Guard requests help to determine the worst case scenario if the ship docks at the port and the item is a nuclear device. Coast Guard requests the RAP and CalOES leads to provide worst case scenarios.
    a. District 11 provides CalOES (California Office of Emergency Services) Situation Report (SITREP) of the details surrounding the suspect ship and material(s).

9. A typical daily flow of information will be inputted into E-wall for (District 11)—this is to simulate the near-real time automatic updates that a network-centric environment would perform. This information will include assets under District 11's OPCON or TACON.

10. RAP and CalOES relay their specific requests for support through the US Navy (Stilleto).

11. The Navy Group accesses reachback capabilities (LLNL- atmospheric modeling reachback and HOPS) to answer the questions about worst case consequences.
    a. This communication should occur in District 11's workspace (Groove).

12. LLNL Atmospheric Modeling provides dose data and HOPS mapping shows areas of critical facilities. They report data back to RAP and CalOES and summarize to the Navy.
    a. This communication should occur in District 11's workspace (Groove).
    b. HOPS mapping site is:
        i. https://hops.llnl.gov/primers/index.html
        ii. Password protected.

13. RAP and CalOES relay the pertinent issues to the Coast Guard.

14. USCG asks Boarding party to review the HOPS mapping site

15. The Boarding Party arrives and boards the target ship. The target ship is searched. They find a radiation signature in two compartments on the ship where the ships documents would indicate no radiation.
    a. The boarding team boards the suspect vessel, and joins the TNT-MIO Boarding Party Workspace in Groove to dialog with the watch team regarding relevant details and the status of action and pass the sensor data into the network for analysis and collaboration.

        i. E-Wall is active to assess its capability (should E-wall notifications go unrecognized, then a phone call or VoIP message must be placed to alert the watch to the situation and E-Wall update).

16. Once the boarding team reports all secure, the Boarding Vessel dialogs with District 11 about departing to proceed on duties assigned (it may remain in the network).

17. The Boarding Party sends radiation data to LLNL reachback for analysis and sends photos of sources.

18. Reachback reports that one signature is NORM – a smoke detector. The second signature indicates U present and with the instruments available it cannot be immediately determined if it is natural or depleted uranium, or whether it is enriched uranium).

19. The Boarding Party checks fingerprints of crewmen (sends data to BFC).
    a. Boarding Team conducts Biometrics. Information is exchanged from the District 11 and MIFC Watch Officers to the Boarding Officer regarding Biometrics. New procedures that reflect the network-centric environment will be enacted to have the Boarding Officer establish a peer-to-peer Groove collaborative workspace with the boarding team, RAP, Radiation reach back (who incorporates LLNL analysts) and BFC. Additionally, SITREPs will be made to District 11 via SA Agent alerts.

    b. THE FOLLOWING EVENTS CAN HAPPEN IN PARALLEL:

20. Obtained data (biometrics, radiological, and visual sensor) and information will be collaborated in Groove with the appropriate centers of excellence for analysis and feedback.
21. Boarding team will report that the ship cannot anchor in the channel and will proceed at slow speed.
22. Three biometrics files will be sent out.
23. One identification comes back as a non-identified person and another comes back with an outstanding warrant.
24. It is reported from the Port of Hong Kong (Austria team insert) that the containers in question were loaded in Hong Kong.
25. The specific shipper in Singapore (Singapore group provides insert – manifest information – requested by Austria) is queried about the contents of the cargo.
26. INJECT: The Shipper (Singapore team reports to Austria team) reports that there was one shipment that was listed as radioactive. It is a medical source of some kind. It is not clear whether the container is properly marked. There is one other container that has a late change in the manifest that may not be on the paperwork that was forwarded.

27. The unresolved shipment is for CalMart Distributors and it says uranium on the manifest.
    a. INJECT: If asked for additional information, then Singapore provides the sender and to whom it is addressed:
        i. Sender: George Koncher
        ii. Addressed to: Committee Against Nuclear Things, Oakland, California
28. INJECT: D11/USCG can submit this web site on the CANT group:
    a. http://www.nrdc.org/nuclear/furanium.asp
29. The Boarding Party confirms that the signature seems to be uranium but they cannot rule out HEU.
    a. Further review of the shipping manifest in Hong Kong reveals one item which was added to the shipment in question right before it was sent to the port that is not normally part of CalMart's shipments.
    b. That item is being sent from George Koncher of the Citizens Against Nuclear Things, an anti-nuclear NGO, from Hong Kong to his home office in California, that item is listed as uranium.
    c. It is decided that this is the continuation of the CANT's efforts to demonstrate the weakness of port security.
    d. The container is flagged and will be searched when off loaded from the vessel.
30. The Boarding Team's initial Identifinder data is taken for 30 seconds only. Radiation reach back will ask for more data. Radiation reach back will dialog with the MSST to acquire the best sensor information upon which to make a determination of what the sources are. Photos will need to be sent to the Export Control advisor.
31. District 11 can ask for a plume calculation from Atmospheric reach back. (If asked, wind speed is 15 mph from the southwest).
32. Once the analysis is complete and the Boarding Officer is made aware of the results, the boarding officer will enter the appropriate alert into SA Agent. (ONCE all biometrics and LLNL analysis reports are returned, mesh network and subordinate elements of the boarding team can prepare for ENDEX).

4.3. Network Operation Events

1. Synchronize clocks between CG District 11, LLNL, YBI TOC and NPS NOC and Stiletto.
2. Set up of **SA** (YBI TOC)
    - Open SA view in computers dedicated to projecting and SA capture
    - Project SW Orion remote view with map on big screen
    - Setup Google Earth SA view
    - Project Google Earth view on big screen
3. Network Applications
    - Verify Groove functional at local NOC

- Verify PELCO viewer for video transmission
- E-WALL
- Camera (V-Stream), Groove, SA Agent and NMS on all participating vessels

4. Setup automatic capturing functionality
   - Verify 2 computers in each location for capture (1 continuous, 1 still)
   - Verify CamStudio and ZapGrab for still capture in SA and SW computers
   - Verify CamStudio and ZapGrab for continuous capture in additional SA computer
5. Review Still Image capture technique
   - Open PowerPoint Producer
   - Click on "Capture" button (cancel out of resultant dialog box)
   - Choose "Still images from screen" then select "Region"
   - Position frame around window(s) of interest
   - Select "Capture Image" and save file in desired directory
6. Continuous screen capture started
   - Open PowerPoint Producer
   - Click on "Capture" button (cancel out of resultant dialog box)
   - Choose "Video screen capture with audio" then "Next"
   - Ensure window(s) of interest inside default frame
   - Select "Capture" to start continuous capture
7. Perform **Configuration Management** functions on Solar Winds (both local and NPS NOC):
   1. Use IP Network Browser to discover the network nodes
   2. Enter manually all discovered network nodes into Solar Winds Monitor view
8. Perform **Fault Management** functions on Solar Winds (both local and NPS NOC):
   - Identify and monitor critical nodes in SW Network Monitor: fault indication, Response time, Packet loss, Node status
   - Install Redline RF Monitoring Tool / monitor RSSI for AN-50 (local NOC) in order to monitor the 802.16 ship-to-ship (YBI /Boarding Vessel – Target Vessel) and ship-to-shore (Boarding Vessel - CG District 11 and Stiletto to MSC) links budget.
   - Critical nodes:
     - o 802.16 ship-to-ship (YBI– Boarding Vessel) link components
     - o 802.16 ship-to-shore (YBI– C2/CG District 11) link components
     - o Biometrics data capture laptop
9. Conduct **Performance Management** functions on Solar Winds (both local on the target vessel and NPS NOC):
   - Select and enter each critical node (interfaces) into Solar Winds Performance Monitor
   - Monitor min/max/average bps in/out, Total bytes transferred, average response time at each interface
   - Monitor Gauges, Real Time Graphs (Throughput)

10. Miscellaneous network management issues:
    - Screen captures during the network operations: real time graphs, gauges, RSSI views, real time monitor (both local and NPS NOC)
    - Manually log events and operational procedures (both local and NPS NOC)
    - Manually log events of Target Vessel (Oakland) radar and communication emissions, course, aspect relative to YBI antenna coverage (YBI TOC)
    - Log any environmental conditions that might have an effect on network operations –i.e. fog, high precipitation (YBI TOC)

5. Measures of Performance

- **Reach-back Performance**
    - Ability of the boarding party to connect and collaborate in order to provide biometric data and Radiation Detection Data via VPN reach back to Biometric Fusion Center and LLNL.
    - Speed and accuracy of radiation detection analyses, multiphased
    - First use of atmospheric reachback in TNT
    - First use of HOPS in TNT, access to web site
    - Access time for remote sites (Operational)
    - Feasibility of applications

- **MIO Collaborative Performance**
    - Latency of sync with all sites (time needed to acknowledge viewing of shared data and status of processing it).
    - Frequency of messaging and ACK
    - Reliability and quality of asset video (remote site observation)
    - Communication protocol within the boarding team
    - Time required for the Boarding Party Members to make decisions to proceed to the next step in the process.
    - Advantages / disadvantages provided by the employed technology to the boarding party in dealing with complex situations such as evaluation of findings on a suspect vessel and specifically the development of a common SA among the participants.
    - Multiple GROOVE discussion windows.

- **Network Performance**
    - Ability to establish 802.16/OFDM links (ship-to-ship and ship-to-shore)
    - Time and feasibility considering the mobility of the target vessel
    - Identified performance variations as a function of geography, geometry, range and electronic interference environment.
    - Throughput as function of time (OFDM)
    - Availability Uplink and Downlink

APPENDIX D. DEFINITIONS

A. NPS MIB

```
NPS-MIB DEFINITIONS ::= BEGIN


IMPORTS
        enterprises,  mgmt,  NetworkAddress,  IpAddress,  Counter,
Gauge,
                TimeTicks
           FROM RFC1155-SMI
        OBJECT-TYPE
                FROM RFC-1212;


-- The NPS MIB, located at 1.3.6.1.4.1.nps
-- Once the NPS enterprise is granted an OID number, that will
-- be entered in place of "nps" above.


nps-mib    OBJECT IDENTIFIER ::= { enterprises (nps) }


--  textual  conventions  (borrowed  from  RFC-1213,  which  doesn't
offer
-- exports)


DisplayString ::=
     OCTET STRING


PhysAddress ::=
     OCTET STRING


-- groups under the subnet-hyper-mib


-- general information within this MIB


general         OBJECT IDENTIFIER ::= { nps-mib 1 }
```

```
nodeIsHypernode OBJECT-TYPE
    SYNTAX  Integer
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
            "A value which indicates the class of Hypernode
            this child represents.

            The value is a sum.  This sum initially takes the
            value zero, Then, for each type, H, in the range
            1 through 3, that this node performs activities
            of, 2 raised to (H - 1) is added to the sum.  For
            example, a node which is only subnetwork aware
            would have a value of 2 (2^(2-1)).  In
            contrast, a node which is both service aware and
            decision support aware would have a value of 5
            (2^(3-1) + 2^(3-1)).  Note that values should be
            calculated accordingly:

                  type  functionality
                     1  Service Aware
                     2  subnetwork aware
                     3  decision support aware
    ::= { general 1 }

END
```

B. Network Service Aware MIB

```
SERVICE-HYPERNODE-MIB DEFINITIONS ::= BEGIN


IMPORTS
        enterprises,  mgmt,  NetworkAddress,  IpAddress,  Counter,
Gauge,
                TimeTicks
             FROM RFC1155-SMI
        OBJECT-TYPE
```

```
                FROM RFC-1212;

--   The   Network   Service   Aware   Hypernode   located   at
1.3.6.1.4.1.nps.1
-- Once the NPS enterprise is granted an OID number, that will
-- be entered in place of "nps" above.

service-hyper-mib    OBJECT IDENTIFIER ::= { enterprises (nps) 1
}

--  textual  conventions  (borrowed  from  RFC-1213,  which  doesn't
offer
-- exports)

DisplayString ::=
     OCTET STRING

PhysAddress ::=
     OCTET STRING

-- groups under the service-hyper-mib

-- provided is a group containing services provided by this node
provided        OBJECT IDENTIFIER ::= { service-hyper-mib 1 }


-- the provided services table.  This is very rudimentary in
-- its present condition.  however, even this table provides
-- useful information for network users.

proServTable OBJECT-TYPE
    SYNTAX  SEQUENCE OF proServEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
       "A list of provided services."
    ::= { provided 1 }

proServEntry OBJECT-TYPE
    SYNTAX  ProServEntry
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
         "An entry in the provided services table describing
          the service provided by this Hypernode"
    INDEX   { proServIndex }
    ::= { proServTable 1 }

          ProServEntry ::=
              SEQUENCE {
                  servIndex
                      INTEGER,

                  servName
                      DisplayString,
                  servReference
                      DisplayString,
                  servDescr
                      DisplayString,
                  servIsMachine
                      BOOLEAN,
                 servIsAvail
                      BOOLEAN,
                  servUrl
                      DisplayString
              }


          servIndex OBJECT-TYPE
              SYNTAX  INTEGER
              ACCESS  read-only
              STATUS  mandatory
              DESCRIPTION
                 "A unique value for each service.  The
                 value for each child must remain constant at
                 least from one re-initialization of the entity's
                 network management system to the next re-
                 initialization, and should remain constant
                 for the life of the node."
              ::= { proServEntry 1 }

          servName OBJECT-TYPE
              SYNTAX  DisplayString (SIZE (0..255))
              ACCESS  read-only
              STATUS  mandatory
              DESCRIPTION
                 "A descriptive name of the service provided."
            ::= { proServEntry 2 }

          servReference OBJECT-TYPE
              SYNTAX  DisplayString (SIZE (0..255))
              ACCESS  read-only
              STATUS  mandatory
              DESCRIPTION
                 "A unique method of referring to this service.
                  This will be relative to the Hypernode network.
                  For instance, this might be the SA ID in TNT."
            ::= { proServEntry 3 }



          servDescr OBJECT-TYPE
              SYNTAX  DisplayString (SIZE (0..255))
              ACCESS  read-only
              STATUS  mandatory
              DESCRIPTION
                 "A free-text description of this service.
```

```
                In   the   absence   of   other   metadata,   this
description
                should be as complete as possible to allow
                other users to make decisions about the use of
                this service."
             ::= { proServEntry 4 }

          servIsMachine OBJECT-TYPE
             SYNTAX  BOOLEAN
             ACCESS  read-only
             STATUS  mandatory

             DESCRIPTION
                "TRUE if this service is provided by automated
                 means.  FALSE if this is a human-provided
                 service."
             ::= { proServEntry 5 }

          servIsAvail OBJECT-TYPE
             SYNTAX  BOOLEAN
             ACCESS  read-only
             STATUS  mandatory

             DESCRIPTION
                "TRUE if this service is currently available.
                Additionally, a myServDown or myServUp trap
                should be sent to appropriate users when the
                Avail status changes."
             ::= { proServEntry 6 }

         servUrl OBJECT-TYPE
             SYNTAX  DisplayString (SIZE (0..255))
             ACCESS  read-only
             STATUS  mandatory
             DESCRIPTION
```

```
                 "The Uniform Resource Locator where the
                 service may be accessed."
              ::= { proServEntry 7 }

-- Traps
-- Trap numbers are reflective of the linkUp, LinkDown numbers,
-- hence the use of traps 2 and 3

nps OBJECT IDENTIFIER ::= { enterprises nps }

myServDown TRAP-TYPE
   ENTERPRISE  nps
    VARIABLES   { servIndex }
    DESCRIPTION
            "A myServDown trap signifies that the sending
            node recognizes a failure in one of the
            services provided by the agent."
     ::= 2

myServUp TRAP-TYPE
   ENTERPRISE  nps
    VARIABLES   { servIndex }
    DESCRIPTION
            "A myServUp trap signifies that the sending
            node recognizes a return to service of one
            of the services provided by the agent."
     ::= 3


END
```

C. Subnetwork Aware MIB

```
SUBNET-HYPERNODE-MIB DEFINITIONS ::= BEGIN

IMPORTS
```

```
        enterprises,  mgmt,  NetworkAddress,  IpAddress,  Counter,
Gauge,
                TimeTicks
           FROM RFC1155-SMI
        OBJECT-TYPE
                FROM RFC-1212;

-- The Subnetwork Aware Hypernode located at 1.3.6.1.4.1.nps.2
-- Once the NPS enterprise is granted an OID number, that will
-- be entered in place of "nps" above.

subnet-hyper-mib OBJECT IDENTIFIER ::= { enterprises (nps) 2 }

--  textual  conventions  (borrowed  from  RFC-1213,  which  doesn't
offer
-- exports)

DisplayString ::=
     OCTET STRING

PhysAddress ::=
     OCTET STRING

-- groups under the subnet-hyper-mib

children   OBJECT IDENTIFIER ::= { subnet-hyper-mib 1 }


childNumber OBJECT-TYPE
    SYNTAX  INTEGER
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
       "The number of child nodes (regardless of
       their current state) present on this subnet."
    ::= { children 1 }
```

```
-- The subnetwork child table

childTable OBJECT-TYPE
    SYNTAX  SEQUENCE OF childEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
       "A list of child nodes.  The number of
       entries is given by the value of childNumber."
    ::= { children 2 }

childEntry OBJECT-TYPE
    SYNTAX  ChildEntry
    ACCESS  read-create
    STATUS  mandatory
    DESCRIPTION
       "A child entry containing layer 3 and 4
       information (IP address, DNS Name, etc.) for a
       given child node as well as some information
      from the system MIB."
    INDEX   { childIndex }
    ::= { childTable 1 }

          ChildEntry ::=
              SEQUENCE {
                  childIndex
                      INTEGER,

                  childName
                      DisplayString,
                  childIp
                      IpAddress,
                  childDescr
                      DisplayString,
             childLocation
```

```
            DisplayString,
        childCached
            BOOLEAN,
             childIsHypernode
                 INTEGER
         }

childIndex OBJECT-TYPE
    SYNTAX  INTEGER
    ACCESS  read-create
    STATUS  mandatory

    DESCRIPTION
      "A unique value for each child.  Its value
      ranges between 1 and the value of childNumber.
The
      value for each child must remain constant at
      least from one re-initialization of the entity's
      network management system to the next re-
      initialization."
    ::= { childEntry 1 }

childName OBJECT-TYPE
    SYNTAX  DisplayString (SIZE (0..255))
    ACCESS  read-create
    STATUS  mandatory
    DESCRIPTION
       "An administratively-assigned name for this
       managed node.  By convention, this is the node's
       fully-qualified domain name."
  ::= { childEntry 2 }

childIp OBJECT-TYPE
    SYNTAX  IpAddress
    ACCESS  read-create
    STATUS  mandatory
    DESCRIPTION
       "The IP address of this child on this subnet.
       In cases where there is more than one IP per
       subnet, the child node will determine which IP
       to advertise."
    ::= { childEntry 3 }

childDescr OBJECT-TYPE
    SYNTAX  DisplayString (SIZE (0..255))
    ACCESS  read-create
    STATUS  mandatory

    DESCRIPTION
      "A textual description of the child.  This value
      should include the full name and version
      identification of the system's hardware type,
      software operating-system, and networking
      software.  It is mandatory that this only contain
      printable ASCII characters."
    ::= { childEntry 4 }

childLocation OBJECT-TYPE
    SYNTAX  DisplayString (SIZE (0..255))
    ACCESS  read-create
    STATUS  mandatory
    DESCRIPTION
       "The physical location of this node (e.g.,
       `telephone closet, 3rd floor')."
    ::= { childEntry 5 }

childCached OBJECT-TYPE
    SYNTAX  BOOLEAN
    ACCESS  read-create
    STATUS  mandatory
```

```
            DESCRIPTION
               "TRUE if we are caching MIB data for this node."
            ::= { childEntry 6 }

        childIsHypernode OBJECT-TYPE
            SYNTAX  INTEGER (0..127)
            ACCESS  read-create
            STATUS  mandatory
            DESCRIPTION
              "A value which indicates the class of Hypernode
              this child represents.

              The value is a sum.  This sum initially takes the
              value zero, Then, for each type, H, in the range
              1 through 3, that this node performs activities
              of, 2 raised to (H - 1) is added to the sum.  For
              example, a node which is only subnetwork aware
              would have a value of 2 (2^(2-1)).  In
              contrast, a node which is both service aware and
              decision support aware would have a value of 5
              (2^(3-1) + 2^(3-1)).  Note that values should be
              calculated accordingly:

                    type  functionality
                       1  Service Aware
                       2  subnetwork aware
                       3  decision support aware

            ::= { childEntry 7 }


-- the child MIB table

childMibTable OBJECT-TYPE
    SYNTAX  SEQUENCE OF childMibEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
       "This table includes all the MIB values for a child."
    ::= { children 3 }

childEntry OBJECT-TYPE
    SYNTAX  ChildMibEntry
    ACCESS  read-create
    STATUS  mandatory
    DESCRIPTION
         "An child entry containing a sequence of
          IP address, OID, MIB Value and description for
          cached MIB values.
    INDEX   { childMibIndex }
    ::= { childMibTable 1 }

          ChildMibEntry ::=
              SEQUENCE {
             childMibIndex
                 INTEGER,
                  childIp
                      IpAddress,
                  childMibOid
                      OBJECT IDENTIFIER,
             childMibValue
                 DisplayString,
             childMibDescr
                 DisplayString,
              }


          childMibIndex OBJECT-TYPE
              SYNTAX  INTEGER
              ACCESS  read-only
              STATUS  mandatory

              DESCRIPTION
                 "A unique value for each cached mib entry."
              ::= { childMibEntry 1 }


          childMibOid OBJECT-TYPE
              SYNTAX  OBJECT IDNETIFIER
              ACCESS  read-only
              STATUS  mandatory

              DESCRIPTION
                      "The OID being cached"
              ::= { childMibEntry 3 }

          childMibValue OBJECT-TYPE
              SYNTAX  DisplayString (SIZE (0..255))
              ACCESS  read-only
              STATUS  mandatory

              DESCRIPTION
                 "A copy of the MIB value from the child node.
                 Since we can't be sure of the syntax for the
                 cached information, we use a display string."
              ::= { childMibEntry 4 }

          childMibDescr OBJECT-TYPE
              SYNTAX  DisplayString (SIZE (0..255))
              ACCESS  read-only
              STATUS  mandatory

              DESCRIPTION
                 "A copy of the MIB description from the child
                 node.  This creates overhead in the case that
```

```
                 this is a standard MIB value, but allows us to
                 be self-describing even for previously unknown
                 OIDs."

              ::= { childMibEntry 5 }

END
```

D. Decision Support Aware MIB

```
DS-HYPERNODE-MIB DEFINITIONS ::= BEGIN

IMPORTS
        enterprises,  mgmt,  NetworkAddress,  IpAddress,  Counter,
Gauge,
                TimeTicks
           FROM RFC1155-SMI
        OBJECT-TYPE
                FROM RFC-1212;

--   The   Decision   Support   Aware   Hypernode   located   at
1.3.6.1.4.1.nps.3
-- Once the NPS enterprise is granted an OID number, that will
-- be entered in place of "nps" above.

ds-hyper-mib    OBJECT IDENTIFIER ::= { enterprises (nps) 3 }

--  textual  conventions  (borrowed  from  RFC-1213,  which  doesn't
offer
-- exports)

DisplayString ::=
     OCTET STRING

PhysAddress ::=
     OCTET STRING

-- groups under the ds-hyper-mib
-- generic
generic               OBJECT IDENTIFIER ::= { ds-hyper-mib 1 }



-- the provided services table.  This is very rudimentary in
-- its present condition.  however, even this table provides
-- useful information for network users.

madeDecisionTable OBJECT-TYPE
    SYNTAX  SEQUENCE OF madeDecisionEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
         "A list of decisions made by or in
          process in this node."
    ::= { generic 1 }

madeDecisionEntry OBJECT-TYPE
    SYNTAX  MadeDecisionEntry
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
         "An entry in the decision table describing
          the decisions made by or outstanding for
          this Hypernode"
    INDEX   { decisionIndex }
    ::= { madeDecisionTable 1 }

          MadeDecisionEntry ::=
              SEQUENCE {
                  decisionIndex
                      INTEGER,

                  decisionName
```

```
            DisplayString,

        decisionDescr
            DisplayString,
   decisionIsOpen
       INTEGER
    }



decisionIndex OBJECT-TYPE
    SYNTAX  INTEGER
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
       "A unique value for each decision.  The
       value for each child must remain constant at
       least from one re-initialization of the entity's
       network management system to the next re-
       initialization, and should remain constant
       longer than the lifecycle of a decision."
    ::= { madeDecisionEntry 1 }

decisionName OBJECT-TYPE
    SYNTAX  DisplayString (SIZE (0..255))
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
       "A descriptive name of the decision.  This
       should be a short but unique identifier of
       the decision."
  ::= { madeDecisionEntry 2 }

decisionDescr OBJECT-TYPE
    SYNTAX  DisplayString (SIZE (0..255))
    ACCESS  read-only
```

```
              STATUS  mandatory
              DESCRIPTION
                 "A free-text description of this decision.
                 In   the   absence   of   other   metadata,   this
description
                 should be as complete as possible to allow
                 other users to take action based on this
                 description."
              ::= { madeDecisionEntry 3 }

          decisionIsOpen OBJECT-TYPE
              SYNTAX  INTEGER {
                other(0),
                new(1),
                open(2),
                awaiting-amplification(3),
                closed-undecided(4),
                closed-decided(5)
                }
              ACCESS  read-only
              STATUS  mandatory

              DESCRIPTION
                 "A numerical representation of the decision
                 state.  New(1) indicates that no action has
                 been taken by the decision maker yet.  Open(2)
                 indicates that the decision maker is working on
                 a decision.  (3) and (5) are self-evident.
                 Closed-undecided(4) indicates that this decision
                 maker  has  chosen  not  to  decide  this.   It  may
have
                 been   passed   to   another   decision-maker   or
tabled."
              ::= { madeDecisionEntry 4 }

-- A Decision inbox table.  As opposed to the read-only nature
-- of the madeDecision table.  Other nodes can modify this table
-- thereby effectively requesting a decision on some subject.

dsInboxTable OBJECT-TYPE
    SYNTAX  SEQUENCE OF dsInboxEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
         "An inbox for requesting decisions."
    ::= { generic 2 }

dsInboxEntry OBJECT-TYPE
    SYNTAX  DsInboxEntry
    ACCESS  read-create
    STATUS  mandatory
    DESCRIPTION
         "An entry in the decision inbox detailing
          the requested decision."

    INDEX   { inboxIndex }
    ::= { dsInboxTable 1 }

          DsInboxEntry ::=
              SEQUENCE {
                  inboxIndex
                      INTEGER,

                  dsInboxName
                      DisplayString,

                  dsInboxDescr
                      DisplayString,
             dsInboxrequester
                 IpAddress
              }
```

```
inboxIndex OBJECT-TYPE
    SYNTAX  INTEGER
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
       "A unique value for each decision.  The
       value for each child must remain constant at
       least from one re-initialization of the entity's
       network management system to the next re-
       initialization, and should remain constant
       longer than the lifecycle of a decision.
    ::= { dsInboxEntry 1 }

dsInboxName OBJECT-TYPE
    SYNTAX  DisplayString (SIZE (0..255))
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
       "A descriptive name of the decision.  This
       should be a short but unique identifier of
       the decision requested."
  ::= { dsInboxEntry 2 }

dsInboxDescr OBJECT-TYPE
    SYNTAX  DisplayString (SIZE (0..255))
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
       "A free-text description of this decision.
       In   the   absence   of   other   metadata,   this
description
       should be as complete as possible to allow
       the decision maker to take for action."
    ::= { dsInboxEntry 3 }
```

```
          dsrequester OBJECT-TYPE
              SYNTAX  IpAddress
              ACCESS  read-only
              STATUS  mandatory

              DESCRIPTION
                 "The IP address of the node requesting
                 the decision."
              ::= { dsInboxEntry 4 }

-- Traps

-- Trap numbers are continuous throughout the nps enterprise
-- OID space we adopt 4 as the next available (after the
-- services MIB traps are considered.

nps OBJECT IDENTIFIER ::= { enterprises nps }

myInboxAccepted TRAP-TYPE
   ENTERPRISE  nps
    VARIABLES   { inboxIndex }
    DESCRIPTION
            "A myDecisionMade trap signifies that the
            sending node has reached a decision."
     ::= 4

myDecisionChangeState TRAP-TYPE
   ENTERPRISE  nps
    VARIABLES   { decisionIndex }
    DESCRIPTION
            "A myDecisionChangeState trap signifies
            that the decisionIsOpen variable has
            changed."
     ::= 5
END
```

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

Abstract Syntax Notation One (ASN.1). Specification of Basic Notation. ITU-T Rec. X.680 (2002) | ISO/IEC 8824-1:2002

Alberts, D., J. Garstka, Richard Hayes and David Signori. Understanding Information Age Warfare. Vienna, Va.: CCRP, 2001.

Bauer, B., and A.S. Patrick (2004). “A Human Factors Extension to the Seven-Layer OSI Reference Model”. Retrieved Jan 6, 2004, from http://www.andrewpatrick.ca/OSI/10layer.html.

Briscoe, B., A. Odlyzko, and B. Tilly. Metcalfe's Law is Wrong. IEEE Spectrum. Volume 43, Issue 7, Jul 2006, pp. 34 - 39

Clark, V. "Sea Power 21: Projecting decisive joint capabilities." Proceedings of the United States Naval Institute 128.10 (Oct 2002):32-.

Clark, F. and M. Hagee. FORCEnet: A Functional Concept for the 21st Century. [www.navy.mil/navydata/policy/forcenet/forcenet21.pdf]. Mar 07.

Clement, M.. Personal conversation. 20 Feb 2006.

Data Processing Open Systems Interconnection – Basic reference Model, Draft Proposal 7498, ISO/TC97/SC 16 N 719, Aug 1981.

Ennis, G, D. Kaufman, and K. Biba, “DoD Protocol Reference Model,” Sytek TR-82026, Sep 1982.

Gateau, J. “Designing Command and Control.” 2006 International Command and Control Research and Technology Symposium: Coalition Command and Control in the Networked Era, Cambridge, England, US Department of Defense, Command and Control Research Program (CCRP), 2006

Glazer, Barney and Anselm Strauss. The Discovery of Grounded Theory: Strategies for Qualitative Research, Aldine de Gruyter, 1967.

Goodhue, Dale. “Understanding User Evaluations of Information Systems.” Management Science, Vol., 41, No. 12. (Dec 1995), pp. 1827-1844.

Goodhue, Dale and Ronal Thompson. “Task-Technology Fit and Individual Performance.” MIS Quarterly, Vol. 19, No. 2. (Jun 1995), pp. 213-236.

Hayes-Roth, F.. “Model-based Communication Networks and VIRT: Filtering Information by Value to Improve Collaborative Decision-Making”. 10th International Command and Control Research and Technology Symposium: The Future of C2, McLean, VA, US Department of Defense, Command and Control Research Program (CCRP), 2005.

Hayes-Roth, F. “Two Theories of Process Design for Information Superiority: Smart Pull vs. Smart Push.” 2006 Command and Control Research and Technology Symposium: The State of the Art and the State of the Practice, San Diego, CA, US Department of Defense, Command and Control Research Program (CCRP), 2006..

Huber, George. “A Theory of the Effects of Advanced Information Technologies on Organizational Design, Intelligence and Decision Making.” The Academy of Management Review, Vol. 15, No. 1 (Jan 1990), pp. 47-71.

Joint Chiefs of Staff. Joint Doctrine for Military Operations Other than War. Joint Publication 3-07. 1995

Lucky, Robert. Silicon Dreams: Information, Man and Machine. New York: St. Martin’s, 1989.

Massink, M. and G. Faconti. “A Reference Framework for Continuous Interaction.” Universal Access in the Information Society. Heidelberg: Aug 2002. Vol. 1, Iss. 4; p. 237.

McArthur, J and Mattis, James. FORCEnet Capabilities Annex. [http://forcenet.navy.mil/concepts/capabilities-annex.pdf]. Mar 07.

“NetOps 100 Student Guide v1.2.2,” Presented by JTF-GNO to The Naval Network and Space Operations Command, Dahlgren, VA, Jan 2005.

Network Working Group Request for Comments: 1157, J. Case, M. Fedor, M. Schoffstall, J. Davin May 1990

Network Working Group K. McCloghrie and M. Rose (eds) Request for Comments: 1213, Mar 1991

NSA Windows XP STIG (12 Sep 2006). [http://www.nsa.gov/snac/downloads_winxp.cfm?MenuID=scg10.3.1.1]. Mar 07.

Oros, Carl (2005). Helicopter Information Awareness Module (I-AM): An example of a Model-Based Communication Network (MCN) Architecture. 10th International Command and Control Research and Technology Symposium: The Future of C2, McLean, VA, US Department of Defense, Command and Control Research Program (CCRP).

Pras, A Drevers, T, v. d. Meent, R, and D. Quartel, “Comparing the Performance of SNMP and Web Services-Based Management,” IEEE eTNSM (Transactions on Network and Service Management), vol. 1, no. 2, 2004.

Romkey, J. “Where I’m Coming From” [http://www.romkey.com/old/from.html]. Mar 07.

Shannon, Claude. The Mathematical Theory of Information. Urbana, University of Illinois Press, 1949.

The Living Internet, “The Internet Toaster” [http://www.livinginternet.com/i/ia_myths_toast.htm]

Von Bertalanffy, Ludwig. General System Theory. New York: George Braziller, 1968.

Wolfowitz, Paul. Department of Defense Instruction 8100.1 19 Sep 2002. Global Information Grid (GIG) Overarching Policy.

Yin, Robert. Case Study Research: Design and Methods, 3rd ed., Sage Publications, 2003.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Dr. Dan Boger
   Naval Postgraduate School
   Monterey, California